**Data Analysis on Spotify Dataset.**

**Name: Venu Sai Ram Udayabhaskara Reddy**

**Introduction:**

The project I am working on is about Spotify music from different genres and their audio characteristics.

SPOTIFY, which offers a huge selection of music and podcast playlists, is one of the most popular online music and podcast streaming services.

We would like to investigate the most popular genres from the entire data set using Apache Spark, Spark SQL, Databricks, Python and Pandas libraries as well as data mining models to see if there is a relationship between genre and volume that would cause a particular genre's song to stand out at the top of the list.

**Data Transformation:**

A diagram of a data cleaning process

Description automatically generated

**Data Collection:**

By using spark API to retrieve metadata, audio characteristics, and other pertinent information about songs.  
Data Preprocessing is the process of organizing and cleaning data. Maintain consistency and handle any missing values.

**Analysis Using Apache Spark and Spark SQL:**

Exploration of Data by using Apache Spark to explore the dataset, comprehend its structure, and extract relevant features. Genre Analysis by using Spark SQL, aggregate and analyze data based on genres to uncover trends and patterns.

**Data Modeling:**

Feature Engineering is the process of creating or extracting significant qualities that may influence a song's success within a genre.

Model Construction to create predictive models, use Python libraries such as Spark, scikit-learn, or Spark's MLlib. Consider regression or classification models to forecast the popularity of a song based on its characteristics within a certain genre.

**Relationship Investigation:**

Using statistical approaches, identify correlations between various auditory properties and the popularity of a genre. Create visual representations (plots, charts, etc.) to explain findings and the correlations between genres and their features.

**Problem Statements:**

* To determine the overall number of original artists?
* How many different genres of music altogether?
* Sentimental analysis on music type?
* Discovered the top five most popular artists?
* The TOP 5 Songs with the Longest Length?
* The song's length in relation to its level of popularity?
* To find loudness of the songs and its popularity?
* Derived Top trending genre?
* Text mining analysis to print the 20 most common words in the album name?

**Details of Dataset:**

By using Spark to read the CSV file into a DataFrame, printing the schema, displaying sample data, and counting the number of columns and records.

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Data Pre-processing:**

The following code is a demonstration of data preprocessing steps in code:

**Dropping Duplicates**: Using dropDuplicates(), remove duplicate rows from the DataFrame.

**Column Renaming**: Change the first column (assuming it's '\_c0') to 'S\_NO'.

**Handling Null Values**: Using fillna(0), fill null values in the DataFrame with zeros.

Finally, after preprocessing, count the number of records to determine how many records remain after removing duplicates and managing null values.

This method cleans the data and prepares the DataFrame for further investigation.

**A screenshot of a computer

Description automatically generated**

**Case-1: Logic To calculate the total number of original artists:**

The function unique\_val appears to be solid! It's intended to count the number of unique values in a given column in DataFrame and display the highest counts for each unique value, specifically for the 'artists' column.

It employs the following procedures:

distinct() and collect() are used to locate unique values in a column.

**Counting Unique Values**: This function counts the number of unique values in a column.

Grouping and Counting: Groups the DataFrame by the provided column and counts the number of occurrences for each unique value in descending order.

**A computer code on a white background

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Case:2 - To discover how many different types of music there are:**

The function, unique\_val('track\_genre'), seeks unique values and counts for the 'track\_genre' column in DataFrame. It appears to be interested in investigating the distinct genres and their occurrence counts within dataset.

Running this function should output the count of unique values for the 'track\_genre' column, followed by the top occurrences of each unique genre, ordered in descending order by count.

This analysis may provide insights into the distribution of various genres within dataset and may identify the most prevalent genres among the songs. Additional steps or modifications to the function may be required if we wish to perform more detailed analysis or investigate links between genres and other variables.

**A screenshot of a computer

Description automatically generated**

**Case:3 - Music type emotional analysis:**

Dividing the 'speechiness' feature into three types: 'Low', 'Medium', and 'High' based on established thresholds. Then, using Matplotlib, count the occurrences of each speechiness type and visualize the median popularity for each type.

The code appears to be well-structured, with the goal of providing an overview of how speechiness kinds relate to median popularity. The following are the steps:

Using withColumn and when conditions, categorize'speechiness' into 'Low,' 'Medium,' and 'High' kinds depending on established thresholds.

Counting Occurrences: Sorting the DataFrame by'speechiness\_type' and counting the number of times each type occurs.

Plotting Median Popularity: Converting the Spark DataFrame to a Pandas DataFrame for visualization with Matplotlib. It appear to be creating a bar chart depicting the median popularity of each speechiness type.

This image will help understand how varying levels of speechiness may affect the popularity of a song.

**A screenshot of a computer code

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Case:4 - Discovered the top five most popular artists:**

To find the top five most popular artists in the DataFrame based on the number of tracks they have. importing this data into the driver, extracting the artist names and counts for plotting, and finally viewing this data with Matplotlib.

Steps are broken out as follows:

Grouping by Artists: Grouping the DataFrame by 'artists' and calculating the number of track occurrences for each artist.

Sorting and restricting: Sorting the artists in descending order by count and restricting the result to the top five artists.

Data Collection: Putting the top five artists' data into the driver and extracting the artist names and counts from the data.

Plotting: Using Matplotlib, generate a horizontal bar chart displaying the track counts for the top five artists. This graphic can quickly show which musicians have the most songs in dataset.

**A screen shot of a computer

Description automatically generated**

**Case:5 - The TOP 5 Songs with the Longest Length:**

Choosing the top five longest songs from the DataFrame based on their duration in milliseconds.Then enter this data into the driver, extract the track names and durations for plotting, and visualize the data with Matplotlib.

The following are the steps:

Column Selection and Filtering: Choosing the 'track\_name' and 'duration\_ms' columns, omitting tracks with the word 'Odeon' in their names, and verifying uniqueness.

Limiting and Sorting: Sorting the data in descending order by 'duration\_ms' and limiting the output to the top five longest tracks.

Data Collection: Entering data from the top five longest songs into the driver.

Extracting Information: Using the obtained data, extract the track names and durations for visualization.

Plotting: Using Matplotlib, generate a horizontal bar chart showcasing the best performers durations.

**A screenshot of a computer

Description automatically generated**

**Case:6 - The song's length in relation to its level of popularity:**

Using Matplotlib to generate a line graph, investigate the association between song duration (in minutes) and popularity in dataframe.

Converting Duration to Minutes: created a new column 'duration\_min' derived from the 'duration\_ms' column to determine the duration of songs in minutes.

Filtering and Choosing Data: Choosing the 'popularity' and 'duration\_min' columns, eliminating tracks with 'Odeon' in their names, verifying uniqueness, and restricting the result to the top ten most popular songs.

Data Collection: Entering the popularity and length of these songs in minutes into the driver.

Extracting Information: Using the acquired data, extracting popularity and duration in minutes for charting.

Plotting: Using Matplotlib, generate a line graph illustrating the association between song duration (in minutes) and popularity.

**A graph with blue lines and dots

Description automatically generated**

**Case:7 - Below is the loudness of the songs and its popularity:**

To use Matplotlib to illustrate the link between song loudness and popularity, concentrating on the top ten most popular songs.

However, the plotting method utilized may not be appropriate for this purpose. The horizontal axis of a bar plot is often used to represent categorical data, whereas loudness and popularity appear to be continuous numerical values.

For the top ten songs based on popularity, plot loudness versus popularity. This visualization could aid in determining whether there is a relationship or trend between the loudness of a song and its popularity.

A graph with blue bars

Description automatically generated A screenshot of a computer

Description automatically generated

**Case:8 - Derived Top trending genre:**

The top trending genres based on popularity will be visualized using Seaborn to create a line plot. Here's how it works:

Top Genres: To choose the 'track\_genre' and 'popularity' columns, sort by 'popularity' in descending order, and get the top 6 rows.

Dropping Duplicates: To ensure originality, duplicates from the top genres based on 'track\_genre' were removed.

Displaying the Outcome: After filtering for top genres, the resulting DataFrame is displayed.

Panda Conversion: For visualization, convert the Spark DataFrame ('trend\_genre') to a Pandas DataFrame ('trend\_genre\_pandas').

Using Seaborn to generate a line plot demonstrating the relationship between genres and their popularity.

A screenshot of a computer code

Description automatically generated

A graph with a line

Description automatically generated

A screen shot of a computer code

Description automatically generated

**Case: 9 - Text mining analysis to print the 20 most common words in the album name:**

Using PySpark, extract and analyze album names to find the most prevalent words. Here is a summary of the steps taken:

Choosing Relevant Columns: Chosen the columns 'track\_name,' 'artists,' and 'album\_name' and discarded rows with missing 'album\_name'.

Tokenizing Album Titles: Breaking down album titles into individual words.

Filtering and cleaning entails removing non-alphabetic characters, stopping words, and only selecting words that contain alphabetic characters.

Counting Word Occurrences: Sorting by word and counting the number of times it appears.

Results: Displaying the 20 most common words found in album titles.

After filtering out common stop words and non-alphabetic letters, this technique identifies the most frequently used terms in album names.

A screenshot of a computer

Description automatically generated

**Conclusion:**

Spotify, as we all know, is a music streaming service that gives a wealth of information about music consumption patterns, such as song popularity, artist popularity, user listening habits, and so on. We were able to acquire insights on many elements of music consumption by examining Spotify datasets, such as:

Popular genres: It is possible to discern trends in music consumption and preferences across different regions and demographics by analyzing the genres of popular songs and singers.

User listening habits: By examining user listening data, patterns in when and how users listen to music, such as time of day, device usage, and listening length, can be identified.

Popular songs and musicians can be identified by evaluating streaming and play count statistics, as well as which songs and artists have had the biggest growth in popularity over time.

These findings can be beneficial to a variety of music business stakeholders, including artists, record labels, and music streaming services themselves. These stakeholders can make better informed decisions about music production, promotion, and distribution if they understand what types of music are popular and how users consume music. However, any conclusions obtained from examining Spotify datasets should be taken with a grain of salt, as the data only reflects a portion of the whole music listening community. Furthermore, Spotify's playlist and suggestion algorithms are private and not entirely transparent, making it impossible to draw definitive conclusions regarding user listening patterns.