**TOP CHARTS OF SPOTIFY FROM 2010-2019**

**A case study report submitted in partial fulfilment of the subject**

**DATA MINING AND DATA WAREHOUSING**

**IN**

**B.TECH. III YEAR VI SEMESTER**

**OF**

**COMPUTER SCIENCE AND ENGINEERING**

**Submitted by**

**HARI ABHISHEK (121710303018)**



**DEPARTMENT OFCOMPUTER SCIENCE & ENGINEERING**

**GITAM INSTITUTE OF TECHNOLOGY**

**GITAM**

**(Deemed to be University)**

**VISAKHAPATNAM**

**APRIL 2020**

**CONTENTS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | Abstract | | | 1 |
|  | Introduction | | | | 5 |
|  | Languages and Associated Libraries/Packages | | | | 6 |
|  | Data Description | | | | 7 |
|  | Methodology and Discussion | | | | 8 |
| 4.1 | | | Data Preprocessing |  |
|  | 4.2 | | | Association Rule Mining |  |
|  | 4.3 | | | Classification Analysis |  |
|  | 4.4 | | | Clustering Analysis |  |
|  | Experimental Results | | | | 15 |
| 5.1 | | <Used techniques names> | |  |
|  | 5.2 | | Apriori and FP-Tree Algorithm | |  |
|  | 5.3 | | ID3 Algorithm | |  |
|  | 5.4 | | < Used technique name> | |  |
|  | Conclusion | | | | 19 |
|  | Uncategorised information | | | | 20 |
|  | References | | | | 22 |

**LIST OF FIGURES**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO.** | **FIGURE NO.** | **FIGURE NAME** | **PAGE NO.** |
| 1  2  3  4  5  6  7 | 5.1.1  5.1.2  5.1.3  5.2.1  5.2.2  5.3  5.4 | **Raw data result**  **Preprocessed data result**  **Preprocessed data heat map result**  **Apriori algorithm returned results**  **FP growth returned results**  **Classification**  **Clustering** | 16  16  17  17  18  18  19 |

**ABSTRACT**

To analyze the changing patterns in the trends and music interests among the people and the composers and directors since 2010. This lets us understand the shift in music culture and audience interest. The data taken into consideration is gathered from the statistics of top charts released by Spotify annually in public interest. The change in genre preference inclusive of their rise and fall can be seen clearly via this data mining.

**MAIN MOTO**

More than 60% of the total time required to complete a data mining project should be spent on data preparation since it is one of the most important contributors to the success of the project. Transforming the data at hand into a format appropriate for knowledge extraction has a signiﬁcant inﬂuence on the ﬁnal models generated, as well as on the amount and quality of the knowledge discovered during the process. At the same time, the eﬀect caused by changes made to a dataset during data preprocessing can either facilitate or complicate even further the knowledge discovery process, thus changes made must be selected with care.

**1. INTRODUCTION**

Music is an instant mood lifter. It touches the soul and helps you connect with people. The music industry has taken a huge leap past the few decades. The recorded **music industry** was **worth** $19.1 billion in 2018, which was almost a double-digit gain (9.7%) from the year Whatever be the genre, it is all equally soothing. Music streaming services are the major players here. Understanding the trends and delivering more of related content is worth a lot for the tech giants like Amazon and Spotify.

This report is organized as follows: Section 1 gives a brief introduction, describes the problem statement and states the objective of the tool. In Section 2, the literature survey is presented followed by Section 3 which deals with the Methodology used in the project. In Section 4, the result has been presented and the report is concluded in section 5.

**1.1 PROBLEM STATEMENT**

There are many genres of music like classical, blues, rock, jazz, folk, etc. Every culture has its own music. The classical music in Indian culture is Carnatic and Hindustani. Music streaming or listening to music online involves understanding the users interests and figuring out what the user wants is a crucial task. The main goal here is to understand the popularity shifts and analyse what genre of songs are trending round the clock so that we get to know how the top charts are being influenced. This tool provides a way to analyse and understand the trends of the music world.

1. **LANGUAGES and ASSOCIATED LIBRARIES/PACKAGES**

To make this possible I have used python as my primary programing language while considering Jupyter notebook and command prompt as the working environments. The collective summation of ML dependant packages and python libraries this has been made possible.

The packages/libraries included for the entire working include the following:

* pandas
* numpy
* pydotplus
* matplotlib.pyplot
* pandas
* apyori
* sklearn.cluster
* sklearn.preprocessing
* sklearn.preprocessing
* seaborn
* matplotlib.pyplot
* sklearn.tree
* sklearn.externals.six
* IPython.display
* sklearn.tree
* scipy

**3. DATA DESCRIPTION**

## **Context**

The top songs BY YEAR in the world by Spotify of the years 2010 to 2019 respectfully. This dataset has several variables about the songs and is based on Billboard. The data here as various constituents of music, which includes the following:

Title: Song's title

Artist: Song's artist

top\_genre: the genre of the track

Year: Song's year in the Billboard

Bpm: Beats.Per.Minute - The tempo of the song.

Nrgy: Energy- The energy of a song - the higher the value, the more energetic song.

Dnce: Danceability - The higher the value, the easier it is to dance to this song.

dB: Loudness/dB - The higher the value, the louder the song

Live: Liveness - The higher the value, the more likely the song is a live recording

Val: Valence - The higher the value, the more positive mood for the song.

Dur: Length - The duration of the song.

Acous: Acousticness - The higher the value the more acoustic the song is.

Spch: Speechiness - The higher the value the more spoken word the song contains.

Pop: Popularity- The higher the value the more popular the song is.

## **Content**

There are the most popular songs in the world by year and 13 variables to be explored. Data were extracted from: [http://organizeyourmusic.playlistmachinery.com](http://organizeyourmusic.playlistmachinery.com/)

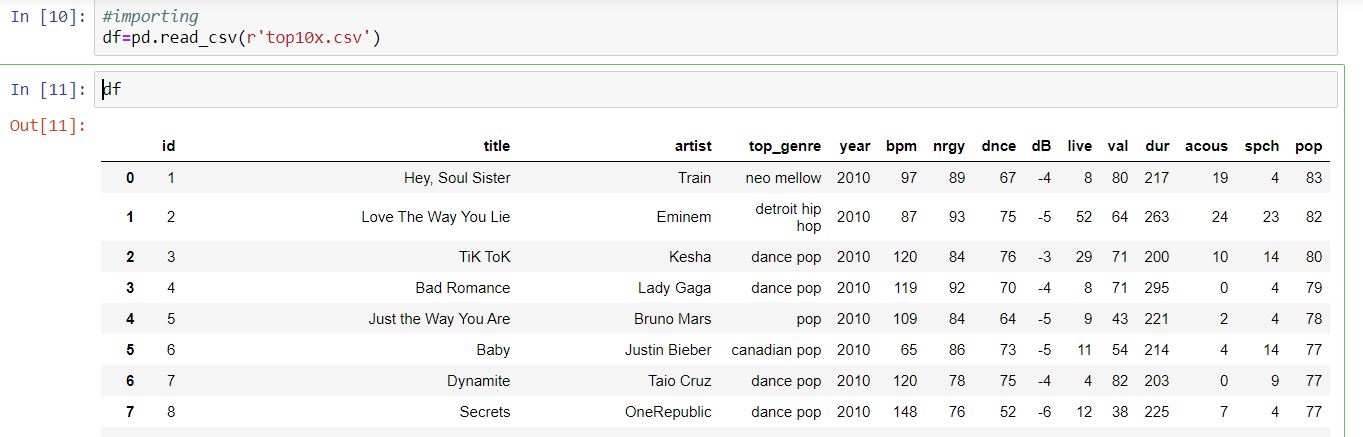
**4. METHODOLOGY AND DISCUSSION**

* + 1. **PREPROCESSING**

1. Today’s real-world datasets are highly susceptible to 
2. noise, missing and inconsistent data due to human errors, 
3. mechanical failures and to their typically large size. Data 
4. aected in this manner is known as “dirty”. During the 
5. past decades, a number of techniques have been developed 
6. to preprocess data gathered from real world applications 
7. before the data is further processed for other purposes. 
8. Cases where data mining techniques are applied 
9. directly to raw data without any kind of data preprocess- 
10. ing are still frequent; yet, data preprocessing has been 
11. recommended as an obligatory step. Data preprocessing 
12. techniques should never be applied blindly to a dataset, 
13. however. Prior to any data preprocessing eort, the dataset 
14. should be explored and characterized. Two methods for 
15. exploring the data prior to preprocessing are data charac- 
16. terization and data visualization
17. Today’s real-world datasets are highly susceptible to 
18. noise, missing and inconsistent data due to human errors, 
19. mechanical failures and to their typically large size. Data 
20. aected in this manner is known as “dirty”. During the 
21. past decades, a number of techniques have been developed 
22. to preprocess data gathered from real world applications 
23. before the data is further processed for other purposes. 
24. Cases where data mining techniques are applied 
25. directly to raw data without any kind of data preprocess- 
26. ing are still frequent; yet, data preprocessing has been 
27. recommended as an obligatory step. Data preprocessing 
28. techniques should never be applied blindly to a dataset, 
29. however. Prior to any data preprocessing eort, the dataset 
30. should be explored and characterized. Two methods for 
31. exploring the data prior to preprocessing are data charac- 
32. terization and data visualization

Real-world datasets are highly susceptible to noise, missing and inconsistent data due to human errors, mechanical failures and to their typically large size. Data affected in this manner is known as “dirty”. During the past decades, a number of techniques have been developed to preprocess data gathered from real world applications before the data is further processed for other purposes. Cases where data mining techniques are applied directly to raw data without any kind of data preprocessing are still frequent, yet data preprocessing has been recommended as an obligatory step.

#importing data:



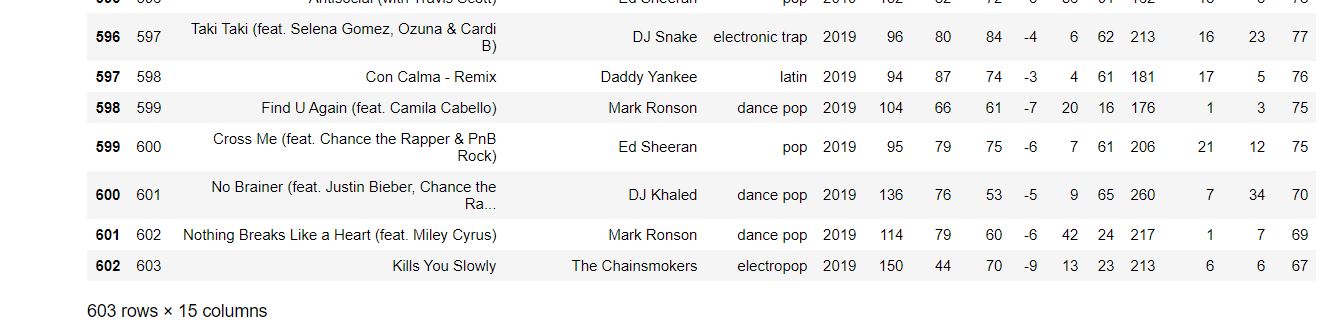
**.**

**.**

**.**

**.**

**.**



**#importing required libraries:**

**import pandas as pd**

**import seaborn as sns**

**import numpy as np**

**import matplotlib.pyplot as plt**

**from matplotlib import rcParams**

**from matplotlib.cm import rainbow**

**from scipy import stats**

**%matplotlib inline**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.preprocessing import StandardScaler**

**train=pd.read\_csv('top10x.csv') #reads file(.csv)**

**print(train)**

**output: 603 rows × 15 columns - Before dropping any rows (fig:5.1.1)**

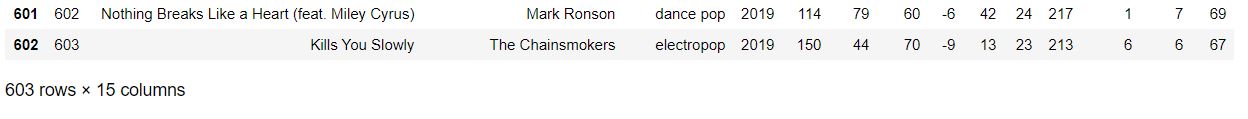
****

**(fig:5.1.1)**

**train1=train.dropna()**

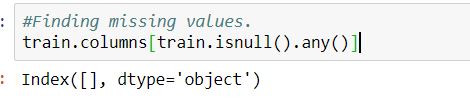
**print(train1)**

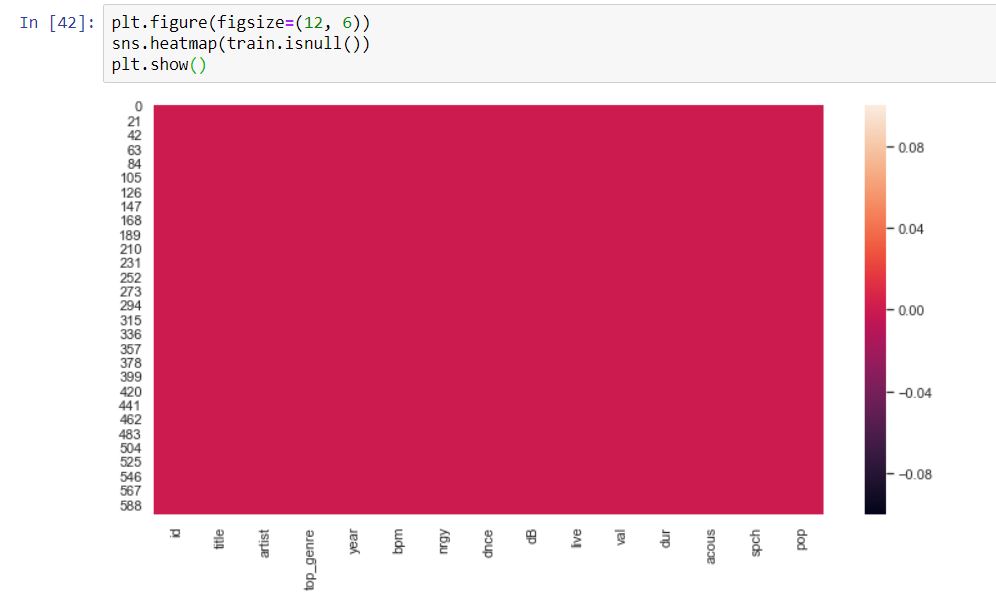
**output: 603 rows × 15 columns After dropping unfit rows(if any)**

**(fig:5.1.2)**

***#Hence there are no null values in the data, as the result pre dropping the rows Vs Post dropping are the same.***

To confirm it we can use the heat map which compares rows and columns on y and x axis respectively. A heat map shows blank lines in between the map if there are missing values, else returns a plain unicolour map.



****

(fig:5.1.3)

*The above processes can be used to identify the null values in a data set taken and can be used to eliminate them accordingly.*

**4.2.** **Association Rule Mining:**

**4.2.1 Apriori Algorithm:**

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

from apyori import apriori

records = []

for i in range(0, num\_records):

records.append([str(store\_data.values[i,j]) for j in range (0, 15)])

store\_data = pd.read\_csv('top10x.csv', header=None)

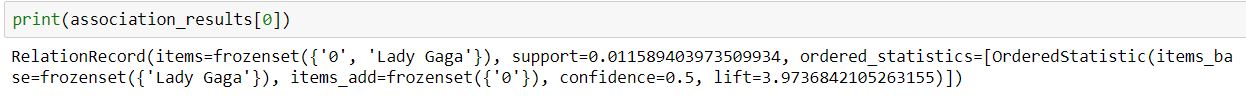
store\_data.head()

num\_records= len(store\_data)

association\_rules = apriori(records, min\_support=0.01, min\_confidence=0.2, min\_lift=3, min\_length=2)

association\_results = list(association\_rules)

***# To check a sample of how the result is shown:***



results=[]

for item in association\_results:

pair = item[0]

items=[x for x in pair]

value0 = str(items[0])

value1 = str(items[1])

value2 = str(item[1])[:7]

value3 = str(item[2][0][2])[:7]

value4 = str(item[2][0][3])[:7]

rows = (value0, value1, value2, value3, value4)

results.append(rows)

Label = ['Title1','Title2','support','confidence','Lift']

store\_suggestion = pd.DataFrame.from\_records(results,columns=Label)

print(store\_suggestion)

Output: **Figure 5.2.1**

**4.2.2 FP Growth Algorithm:**

\*pip install pyfpgrowth\*

import numpy as np

import pandas as pd

import pyfpgrowth

data = pd.read\_csv(r'top10x.csv')

x, y = data.shape

records = []

for i in range(x):

records.append([str(data.values[i,j]) for j in range(y)])

patterns = pyfpgrowth.find\_frequent\_patterns(records, 2)

rules = pyfpgrowth.generate\_association\_rules(patterns, 0.7)

print(rules)

Output: **Figure 5.2.2**

**4.3 Classification:**

import pandas as pd

import pydotplus

from sklearn.tree import DecisionTreeClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn import metrics

from sklearn.preprocessing import LabelEncoder

from IPython.display import Image

from sklearn.externals.six import StringIO

from graphviz import Source

from sklearn import tree

from sklearn.tree import export\_graphviz

from sklearn.externals.six import StringIO

from IPython.display import Image

from sklearn.tree import DecisionTreeClassifier

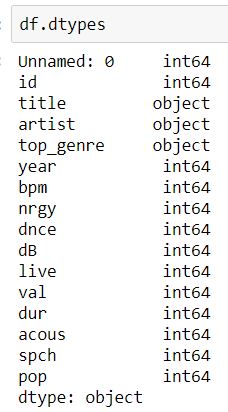
import os

os.environ['PATH'] = os.environ['PATH']+';'+os.environ['CONDA\_PREFIX']+r"\Library\bin\graphviz"

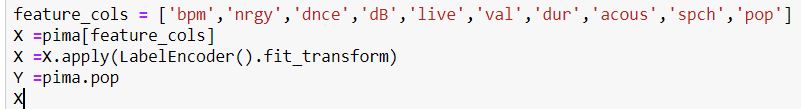
df = pd.read\_csv("top10x.csv")

df.head()

df.Day = df.year.astype(object)



pandas\_df=spark\_df.toPandas()



X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.1, random\_state=1)

clf = DecisionTreeClassifier()

clf = clf.fit(X\_train,Y\_train)

y\_pred = clf.predict(X\_test)

print("Accuracy:",metrics.accuracy\_score(Y\_test, y\_pred))

dt=DecisionTreeClassifier(class\_weight="balanced", min\_samples\_leaf=30)

fit\_decision=dt.fit(X,Y)

dot\_data = StringIO()

export\_graphviz(dt, out\_file=dot\_data,

filled=True, rounded=True,

special\_characters=True,feature\_names = feature\_cols)

graph = pydotplus.graph\_from\_dot\_data(dot\_data.getvalue())

Image(graph.create\_png())

Output: **Figure 5.3**

**4.4 Clustering:**

import pandas as pd

import numpy as np

from sklearn.cluster import KMeans

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import MinMaxScaler

import seaborn as sns

import matplotlib.pyplot as plt

%matplotlib inline

ds=pd.read\_csv(r"top10x.csv", usecols=['year','title','artist','top\_genre','bpm', 'nrgy', 'dnce', 'dB','live', 'val', 'dur', 'acous', 'spch','pop'])

ds.head()

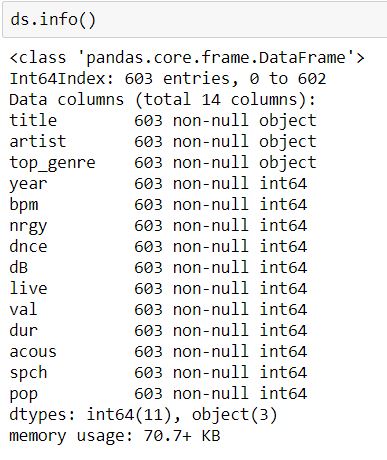
ds = ds.dropna()

ds.head()

print(ds.columns.values)

ds.isna().head()

ds[['pop','top\_genre']].groupby(['top\_genre'], as\_index=False).mean().sort\_values(by='pop', ascending=False)



labelEncoder = LabelEncoder()

labelEncoder.fit(ds['year'])

labelEncoder.fit(ds['title'])

labelEncoder.fit(ds['artist'])

labelEncoder.fit(ds['top\_genre'])

labelEncoder.fit(ds['nrgy'])

labelEncoder.fit(ds['val'])

labelEncoder.fit(ds['dur'])

labelEncoder.fit(ds['pop'])

labelEncoder.fit(ds['spch'])

labelEncoder.fit(ds['live'])

labelEncoder.fit(ds['dnce'])

labelEncoder.fit(ds['bpm'])

ds['year'] = labelEncoder.fit\_transform(ds['year'])

ds['title'] = labelEncoder.fit\_transform(ds['title'])

ds['artist'] = labelEncoder.fit\_transform(ds['artist'])

ds['top\_genre'] = labelEncoder.fit\_transform(ds['top\_genre'])

ds['nrgy'] = labelEncoder.fit\_transform(ds['nrgy'])

ds['val'] = labelEncoder.fit\_transform(ds['val'])

ds['dur'] = labelEncoder.fit\_transform(ds['dur'])

ds['pop'] = labelEncoder.fit\_transform(ds['pop'])

ds['spch'] = labelEncoder.fit\_transform(ds['spch'])

ds['live'] = labelEncoder.fit\_transform(ds['live'])

ds['dnce'] = labelEncoder.fit\_transform(ds['dnce'])

ds['bpm'] = labelEncoder.fit\_transform(ds['bpm'])

X = np.array(ds.drop(['top\_genre'], 1).astype(int))

y = np.array(ds['top\_genre'])

kmeans = KMeans(n\_clusters = 4, max\_iter=100, algorithm = 'auto')

scaler = MinMaxScaler()

X\_scaled = scaler.fit\_transform(X)

kmeans.fit(X\_scaled)

correct = 0

for i in range(len(X)):

predict\_me = np.array(X[i].astype(float))

predict\_me = predict\_me.reshape(-1, len(predict\_me))

prediction = kmeans.predict(predict\_me)

if prediction[0] == y[i]:

correct += 1

print(correct/len(X))

res = kmeans.fit\_predict(X)

print(res)

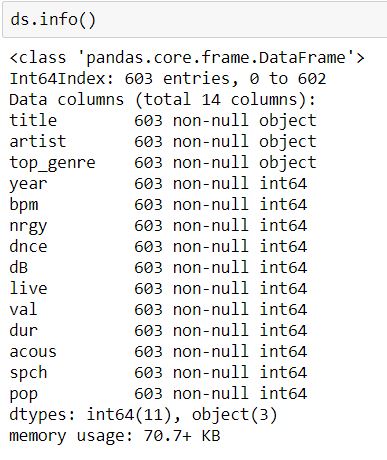
plt.scatter(X[res == 0, 0], X[res == 0, 1], s=50, c='red', marker='s', edgecolor='black', label='cluster 1')

plt.scatter(X[res == 1, 2], X[res == 1, 1])

**Output: Figure 5.4**

1. **EXPERIMENTAL RESULTS**

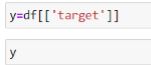
**RESULTS FROM DATA-PREPROCESSING:**

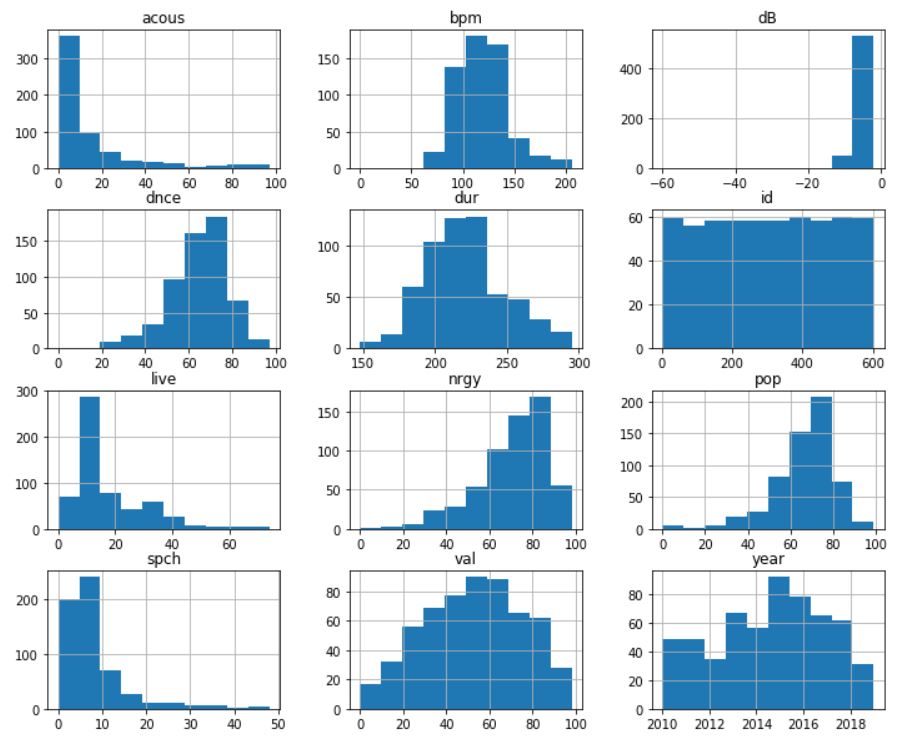
****

**From the output above, there are a total of 13 features and 1 target variable. Also, there are no missing values. Next, the describe() method is used.**

****

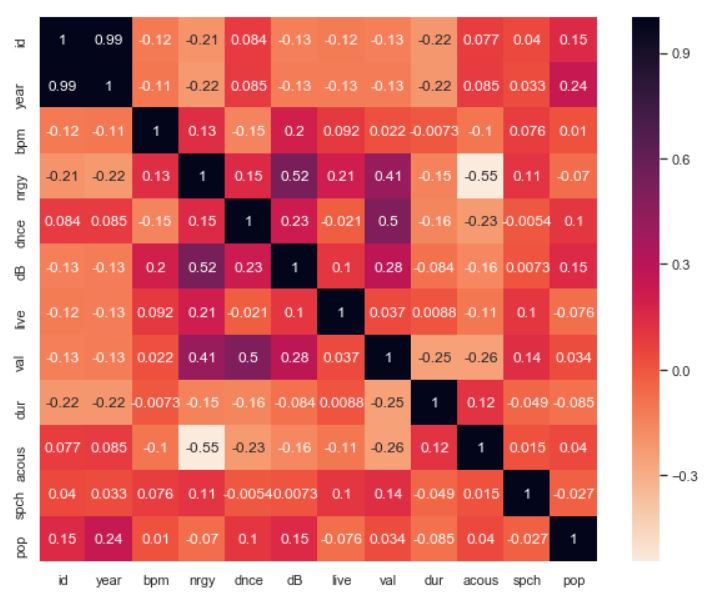
**Generating Histogram:**

****

****

**It shows how each feature and label is distributed along different ranges, which further confirms the need for scaling. Next, wherever you see discrete bars, it basically means that each of these is actually a categorical variable. We will need to handle these categorical variables before applying Machine Learning.**

**#Correlation Matrix:**

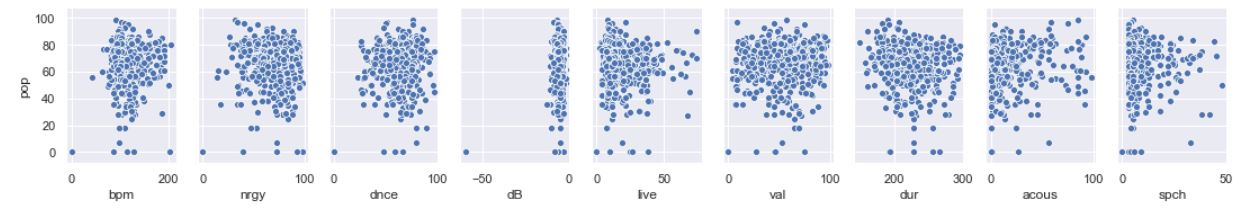
****

**#** **Correlation between popularity and the other numeric features:**

**g = sns.pairplot(df, y\_vars="pop", x\_vars=['bpm', 'nrgy', 'dnce', 'dB','live', 'val', 'dur', 'acous', 'spch'])**

**g.fig.set\_figheight(3)**

**g.fig.set\_figwidth(15)**

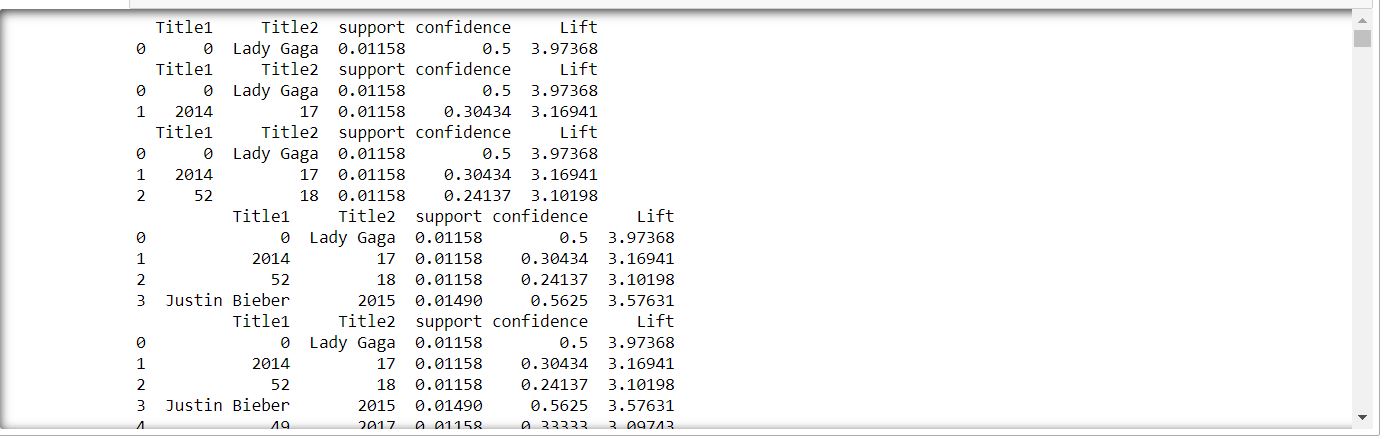
****

**RESULTS AFTER ASSOCIATION RULE MINING:**

**APRIORI:**

**Above we have already seen the rules, packages and implementation of apriori algorithm.**

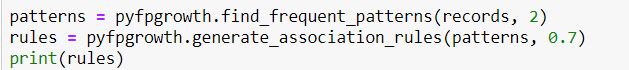
****

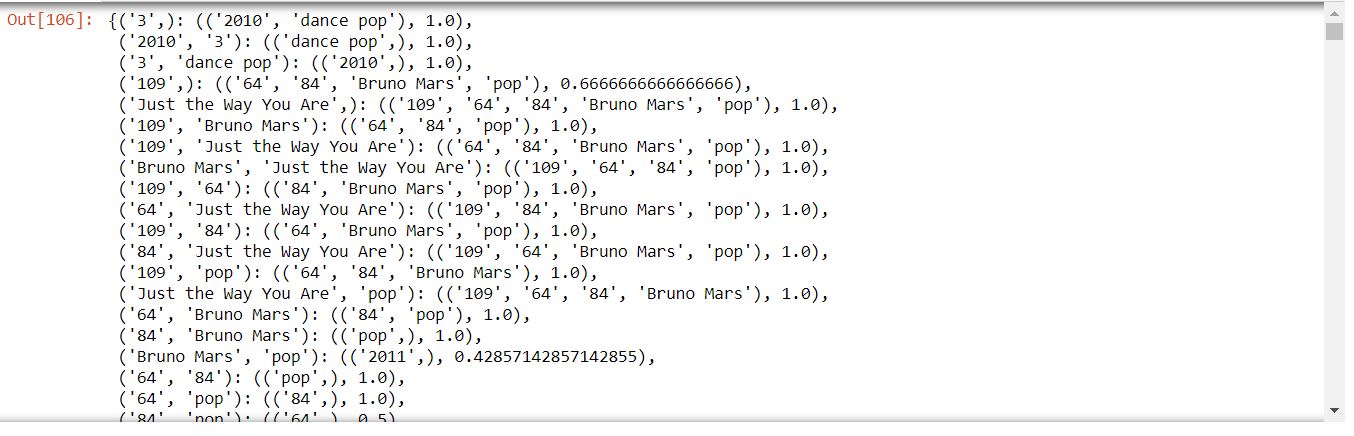
****

**Figure 5.2.1: Apriori algorithm returned results**

**FP-GROWTH TREE:**

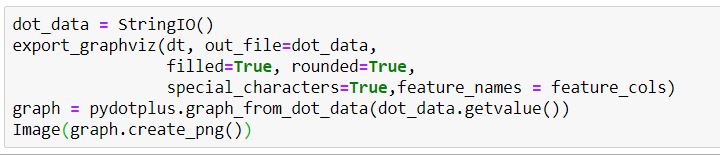
**The rules for FP growth tree and the implementation are shown above. The final patterns and rules are:**

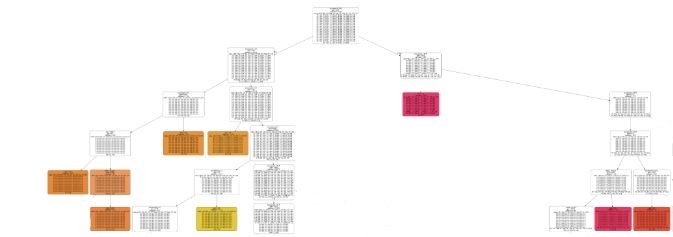
****

****

**Figure 5.2.2: FP growth algorithm returned results**

**RESULTS AFTER CLASSIFICATION:**

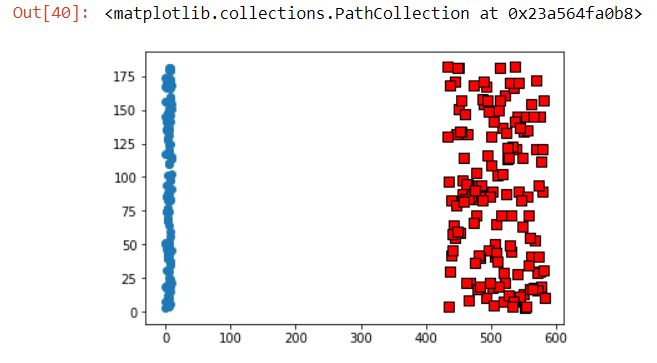
****

****

**Figure 5.3: Clasification**

**CLUSTERING RESULTS:**

****

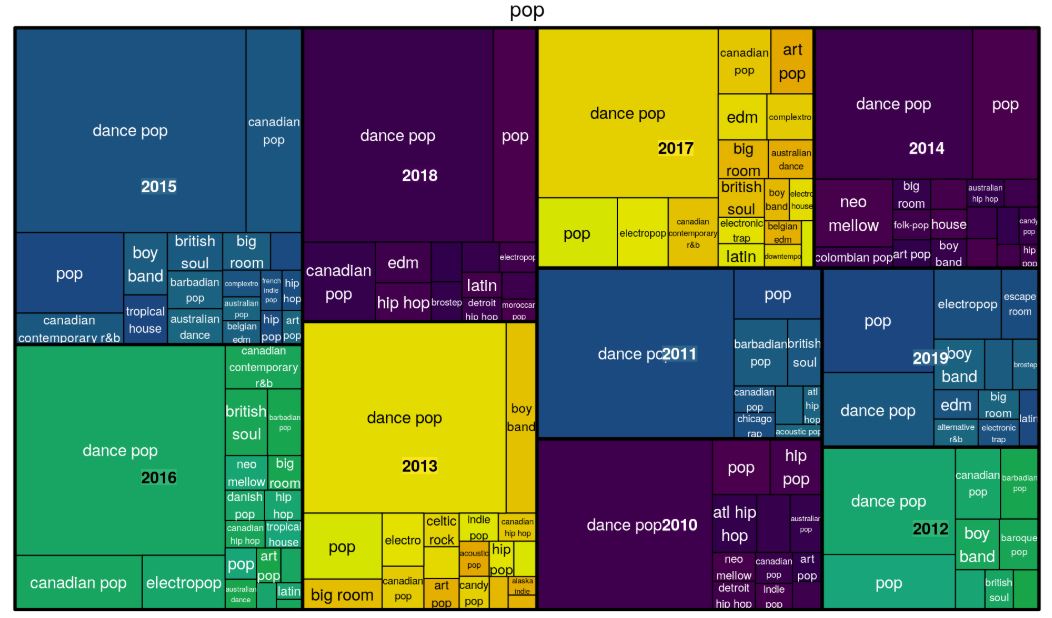
****

**Figure 5.4: clustering result**

**6. CONCLUSION**

**The interests of the audience and the domain experts have been changing by a swing since a while. It is clearly seen that from 2010 to 2019 the genre and artists have been disappearing and popularity levels have been widely increased in accordance with the other constituent factors.**

**The following is the pictographic representation of it:**

****

**7. REFERENCES**

1. International Encyclopedia of Statistical Science, pp.1083-1085
2. Lesson Plan in Glearn where python libraries are mentioned at the end of the each module.
3. <https://www.geeksforgeeks.org/data-mining/>
4. <https://stackoverflow.com/search?q=data+mining>
5. <https://cmdlinetips.com/>
6. <https://towardsdatascience.com/predicting-presence-of-heart-diseases-using-machine-learning-36f00f3edb2c>
7. https://stackabuse.com/association-rule-mining-via-apriori-algorithm-in-python/

**\*\*\*\*\*\*\*\*\***