19ZO02-Social and Economic Network Analysis

Project Report

Topic: Genre and Popularity Analysis of Songs

Team Members:

19Z226	M.MANOJKUMAR
19Z228	NANDHA KISHORE. V
19Z234	RAHUL RAJ. D
19Z236	RASWANTH. E. A
19Z242	SANJAI S

BACHELOR OF ENGINEERING

Branch: COMPUTER SCIENCE AND ENGINEERING

Of Anna University

KNOWLEDGE AND SERVICE

1951

PSG College of Technology

Coimbatore – 641004

1. Problem Statement:

Objective is to analyze genre, artists and other attributes based on popularity and predict the popularity using other attributes in dataset and genre using Artists. For this we make use of dataset from Kaggle for representing the song name, artists, genre, popularity and other attributes related to songs as a graph with

- Artists and Genre serve as Nodes
- Songs serves as Edges.
- The Edges are weighed according to the popularity of genre and artists of songs.

2. Dataset Description:

- The dataset is comprised of song names, genre, artists and popularity of the songs.
- This dataset was extracted from the Kaggle database.
- Dataset Attributes:
    - index - Row ID
    - Title – song name
    - Artists
    - Genre – Type of song
    - Popularity
    - Danceability, valence, loudness, liveness, energy, acoustic, length,
- Dataset: https://www.kaggle.com/datasets/iamsumat/spotify-top-2000s-mega-dataset

3. Tools **used:**

- **Python:** We have used the Python Language for the coding part because of its User-friendly Data Structures.
- **Google Collab**: Google Collab is particularly well suited to machine learning, data analysis, and education since it enables anyone to develop and run arbitrary Python code through the internet. Python code may be written and run through a browser using Google Collab.
- **Packages used:** matplotlib, sklearn, pandas, cborn, networkx.

4.Challenges Faced:

- There was some error in the code, so we were unable to visualize the graph Initially.
- Even though our code was debugged and ran, the expected output in terms of degree and centrality were all 0.
- Since we were new to Network X, it was tiring to understand and visualize the graphs.

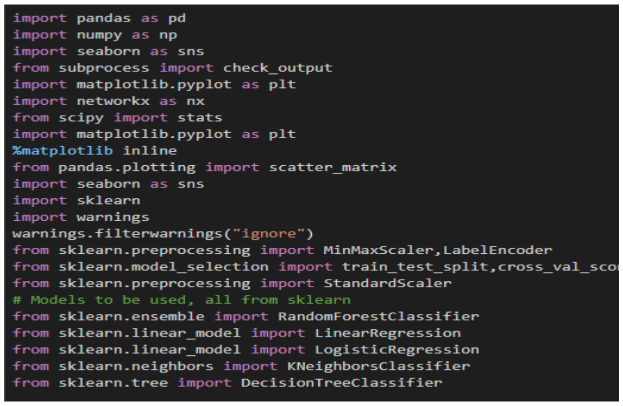
5.Contribution:

Name (Roll number)	**Contribution**
M.Manojkumar(19Z226)	Graph Visualization, Analysis and coding
Nandha Kishore V (19Z228)	Collecting dataset and Preprocessing
Rahul Raj D (19Z226)	Documentation and Statistical analysis
Raswanth E A(19Z236)	Graph Visualization
Sanjai S (19Z242)	Dataset Analysis

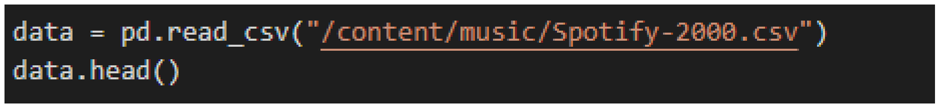
Annexure – I:

https://github.com/MManojkumar16/SENA-PROJECT

Annexure- II:

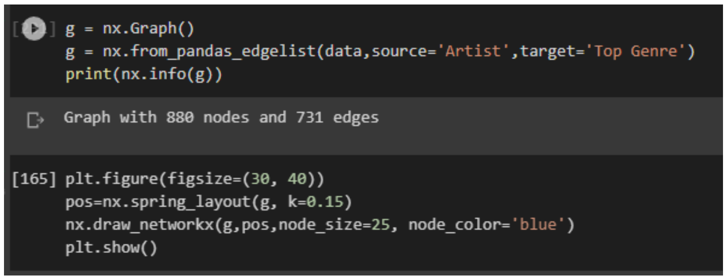


```
import pandas as pd
import numpy as np
import seaborn as sns
from subprocess import check_output
import matplotlib.pyplot as plt
import networkx as nx
from scipy import stats
import matplotlib.pyplot as plt
%matplotlib inline
from pandas.plotting import scatter_matrix
import seaborn as sns
import sklearn
import warnings
warnings.filterwarnings("ignore")
from sklearn.preprocessing import MinMaxScaler,LabelEncoder
from sklearn.model_selection import train_test_split,cross_val_sco
from sklearn.preprocessing import StandardScaler
# Models to be used, all from sklearn
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
```



```
data = pd.read_csv("/content/music/Spotify-2000.csv")
data.head()
```

Figure 1: Loading the Libraries and describing the dataset information



```
[ ] g = nx.Graph()
    g = nx.from_pandas_edgelist(data,source='Artist',target='Top Genre')
    print(nx.info(g))

    Graph with 880 nodes and 731 edges

[165] plt.figure(figsize=(30, 40))
      pos=nx.spring_layout(g, k=0.15)
      nx.draw_networkx(g,pos,node_size=25, node_color='blue')
      plt.show()
```

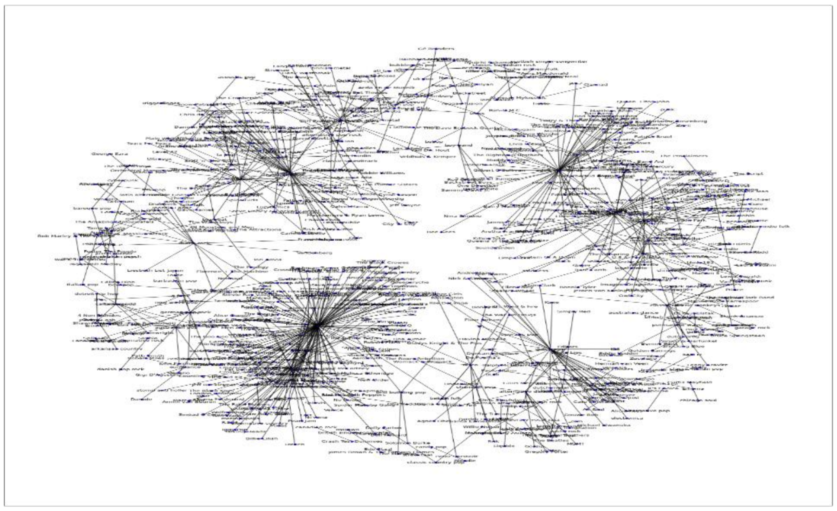
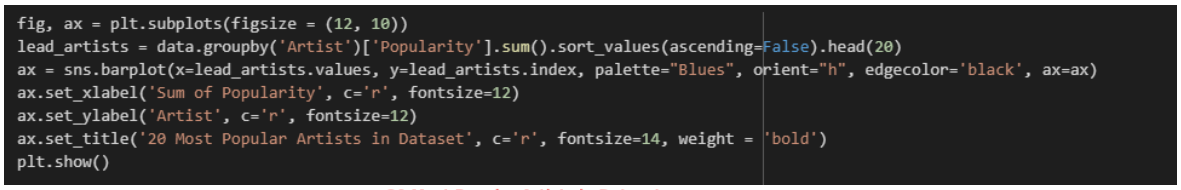


Figure 2: Shows how Artists and Genre are related by using Network Graph.



```
fig, ax = plt.subplots(figsize = (12, 10))
lead_artists = data.groupby('Artist')['Popularity'].sum().sort_values(ascending=False).head(20)
ax = sns.barplot(x=lead_artists.values, y=lead_artists.index, palette="Blues", orient="h", edgecolor='black', ax=ax)
ax.set_xlabel('Sum of Popularity', c='r', fontsize=12)
ax.set_ylabel('Artist', c='r', fontsize=12)
ax.set_title('20 Most Popular Artists in Dataset', c='r', fontsize=14, weight = 'bold')
plt.show()
```

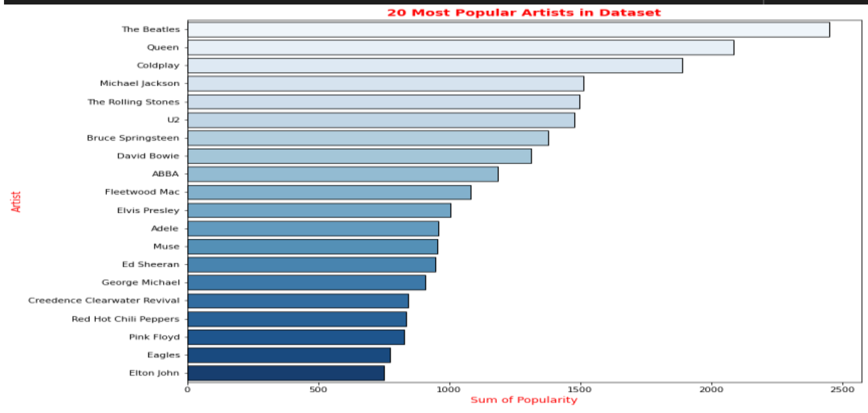
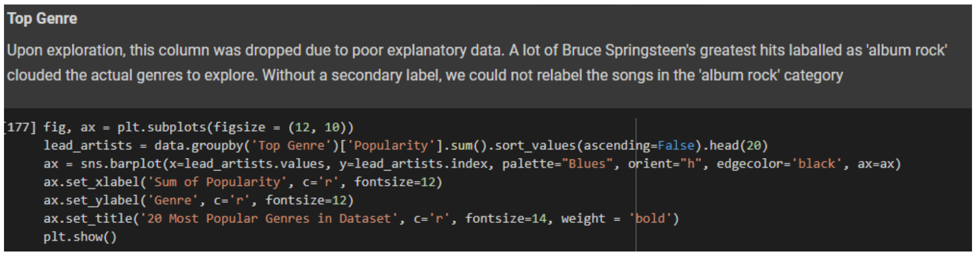


Figure 3: Shows 20 Most Popular Artists in the dataset.



Top Genre

Upon exploration, this column was dropped due to poor explanatory data. A lot of Bruce Springsteen's greatest hits laballed as 'album rock' clouded the actual genres to explore. Without a secondary label, we could not relabel the songs in the 'album rock' category

```
[177] fig, ax = plt.subplots(figsize = (12, 10))
      lead_artists = data.groupby('Top Genre')['Popularity'].sum().sort_values(ascending=False).head(20)
      ax = sns.barplot(x=lead_artists.values, y=lead_artists.index, palette="Blues", orient="h", edgecolor='black', ax=ax)
      ax.set_xlabel('Sum of Popularity', c='r', fontsize=12)
      ax.set_ylabel('Genre', c='r', fontsize=12)
      ax.set_title('20 Most Popular Genres in Dataset', c='r', fontsize=14, weight = 'bold')
      plt.show()
```

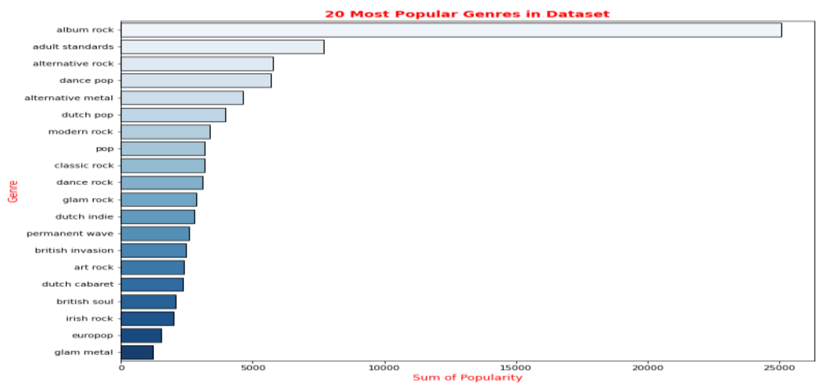
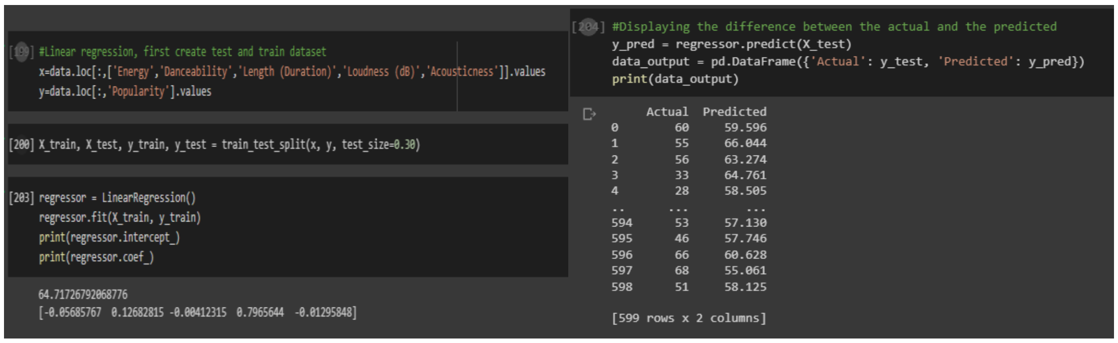


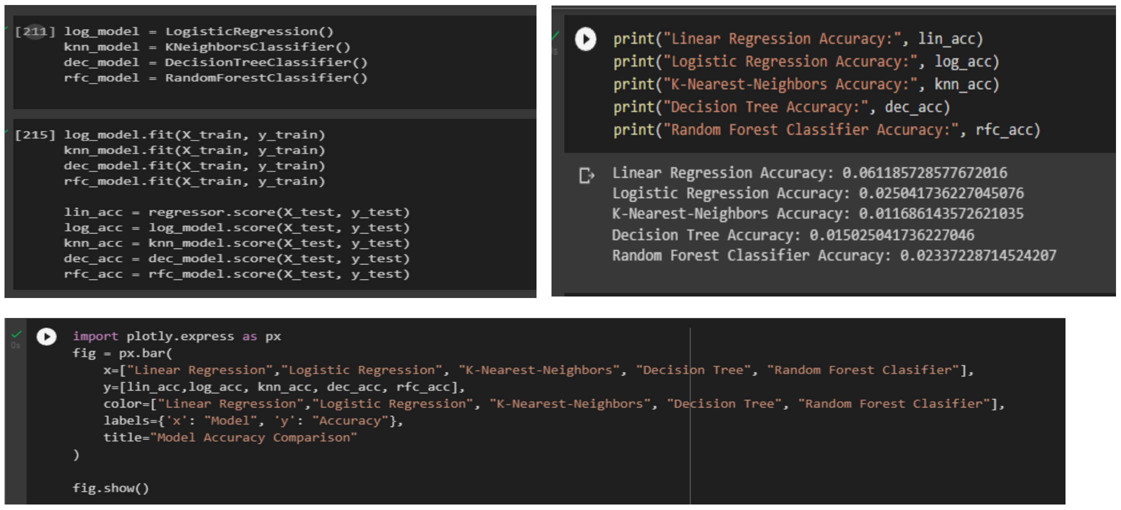
Figure 4: Shows 20 Most Popular Genres in the Dataset.



```
[199] #Linear regression, first create test and train dataset
      x=data.loc[:,['Energy','Danceability','Length (Duration)','Loudness (dB)','Acousticness']].values
      y=data.loc[:,'Popularity'].values

[200] X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.30)

[203] regressor = LinearRegression()
      regressor.fit(X_train, y_train)
      print(regressor.intercept_)
      print(regressor.coef_)

      64.71726792068776
      [-0.05685767  0.12682815 -0.00412315  0.7965644  -0.01295848]
```

```
[204] #Displaying the difference between the actual and the predicted
      y_pred = regressor.predict(X_test)
      data_output = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
      print(data_output)

         Actual  Predicted
      0      60     59.596
      1      55     66.044
      2      56     63.274
      3      33     64.761
      4      28     58.505
      ..    ...        ...
      594    53     57.130
      595    46     57.746
      596    66     60.628
      597    68     55.061
      598    51     58.125

      [599 rows x 2 columns]
```

Figure 5: Shows the Prediction of Popularity using Linear Regression Model.



```
[211] log_model = LogisticRegression()
      knn_model = KNeighborsClassifier()
      dec_model = DecisionTreeClassifier()
      rfc_model = RandomForestClassifier()

[215] log_model.fit(X_train, y_train)
      knn_model.fit(X_train, y_train)
      dec_model.fit(X_train, y_train)
      rfc_model.fit(X_train, y_train)

      lin_acc = regressor.score(X_test, y_test)
      log_acc = log_model.score(X_test, y_test)
      knn_acc = knn_model.score(X_test, y_test)
      dec_acc = dec_model.score(X_test, y_test)
      rfc_acc = rfc_model.score(X_test, y_test)
```

```
print("Linear Regression Accuracy:", lin_acc)
print("Logistic Regression Accuracy:", log_acc)
print("K-Nearest-Neighbors Accuracy:", knn_acc)
print("Decision Tree Accuracy:", dec_acc)
print("Random Forest Classifier Accuracy:", rfc_acc)

Linear Regression Accuracy: 0.061185728577672016
Logistic Regression Accuracy: 0.025041736227045076
K-Nearest-Neighbors Accuracy: 0.011686143572621035
Decision Tree Accuracy: 0.015025041736227046
Random Forest Classifier Accuracy: 0.02337228714524207
```

```
import plotly.express as px
fig = px.bar(
    x=["Linear Regression","Logistic Regression", "K-Nearest-Neighbors", "Decision Tree", "Random Forest Clasifier"],
    y=[lin_acc,log_acc, knn_acc, dec_acc, rfc_acc],
    color=["Linear Regression","Logistic Regression", "K-Nearest-Neighbors", "Decision Tree", "Random Forest Clasifier"],
    labels={'x': "Model", 'y': "Accuracy"},
    title="Model Accuracy Comparison"
)

fig.show()
```

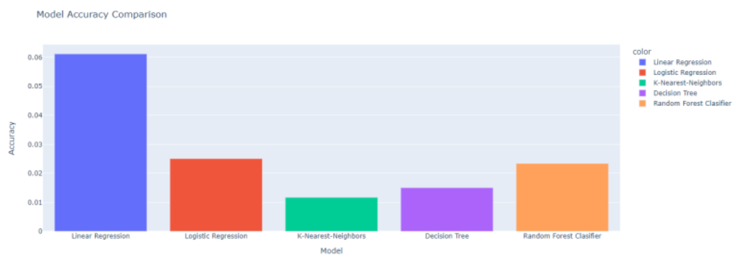
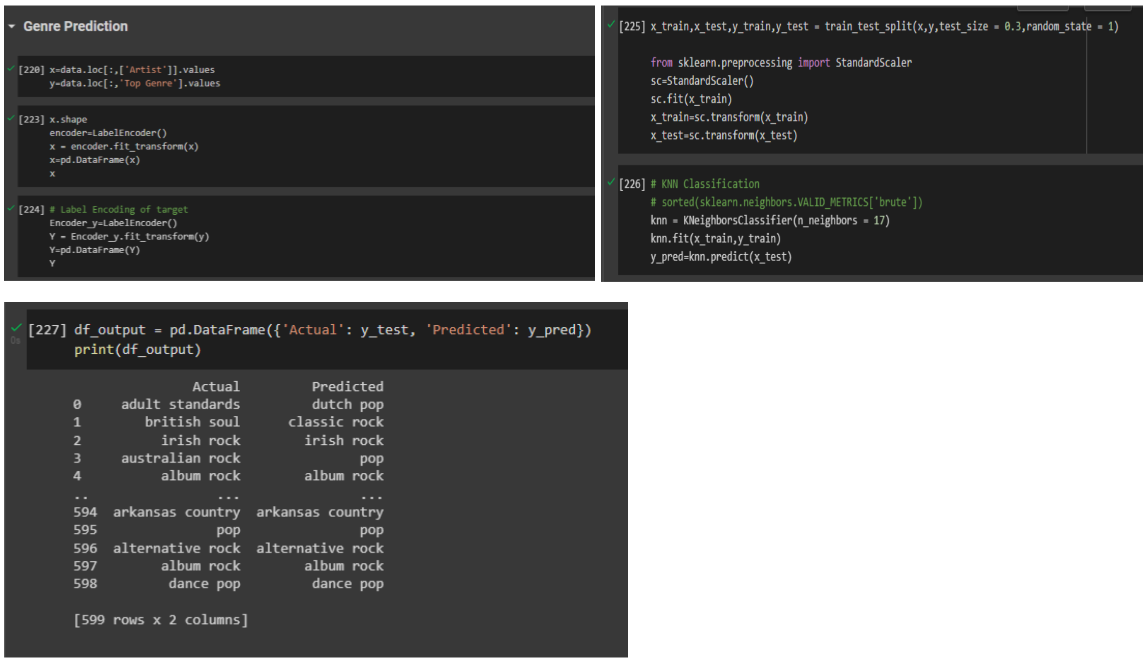


Figure 6: Shows the Predictive accuracy of popularity using all other modules and displaying the same.



```
[220] x=data.loc[:,['Artist']].values
      y=data.loc[:,'Top Genre'].values

[223] x.shape
      encoder=LabelEncoder()
      x = encoder.fit_transform(x)
      x=pd.DataFrame(x)
      x

[224] # Label Encoding of target
      Encoder_y=LabelEncoder()
      Y = Encoder_y.fit_transform(y)
      Y=pd.DataFrame(Y)
      Y
```

```
[225] x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.3,random_state = 1)

      from sklearn.preprocessing import StandardScaler
      sc=StandardScaler()
      sc.fit(x_train)
      x_train=sc.transform(x_train)
      x_test=sc.transform(x_test)

[226] # KNN Classification
      # sorted(sklearn.neighbors.VALID_METRICS['brute'])
      knn = KNeighborsClassifier(n_neighbors = 17)
      knn.fit(x_train,y_train)
      y_pred=knn.predict(x_test)
```

```
[227] df_output = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
      print(df_output)

                 Actual         Predicted
     0     adult standards         dutch pop
     1        british soul      classic rock
     2          irish rock        irish rock
     3     australian rock               pop
     4          album rock        album rock
     ..                ...               ...
     594  arkansas country  arkansas country
     595               pop               pop
     596  alternative rock  alternative rock
     597        album rock        album rock
     598         dance pop         dance pop

     [599 rows x 2 columns]
```

Figure 7: Shows the Genre Prediction based on Artists using KNN Classification Model.

References:

[1] Kaggle Dataset — https://www.kaggle.com/datasets/iamsumat/spotify-top-2000s-mega-dataset

[2] NetworkX —https://networkx.org/documentation/stable/reference/algorithms/index.html

[3] Pandas- https://pandas.pydata.org/

[4] https://medium.com/web-mining-is688-spring-2021/network-of-genre-and-artists-in-spotify-98c896569dee

[5] https://arxiv.org/abs/1706.01953

[6] https://ieeexplore.ieee.org/document/9675998

[7] https://dl.acm.org/doi/10.1145/3474085.3475495