Spotify Genre Prediction

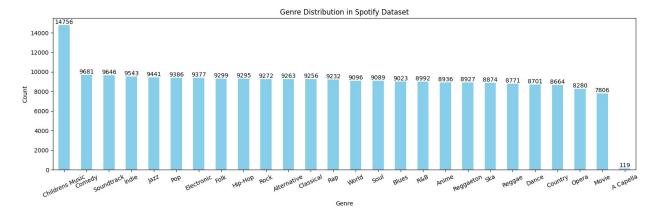
Source

```
import pandas as pd
import matplotlib.pyplot as plt
df raw = pd.read csv('SpotifyFeatures.csv')
df raw.head()
                                                     track name
   genre
                artist name
  Movie
             Henri Salvador
                                   C'est beau de faire un Show
  Movie Martin & les fées
                              Perdu d'avance (par Gad Elmaleh)
                                Don't Let Me Be Lonely Tonight
  Movie
            Joseph Williams
                                Dis-moi Monsieur Gordon Cooper
  Movie
             Henri Salvador
   Movie
               Fabien Nataf
                                                      Ouverture
                 track id
                            popularity
                                        acousticness
                                                       danceability \
   OBRj06ga9RKCKjfDgeFgWV
                                                0.611
                                                              0.389
1
  OBjC1NfoE00usryehmNudP
                                     1
                                                0.246
                                                              0.590
   OCoSDzoNIKCRs124s9uTVv
                                     3
                                                0.952
                                                              0.663
3
   OGc6TVm52BwZD07Ki6tIvf
                                     0
                                                0.703
                                                              0.240
  0IuslXpMROHdEPvSl1fTQK
                                     4
                                                0.950
                                                              0.331
                energy instrumentalness key
                                               liveness
   duration ms
                                                          loudness
mode \
         99373
                 0.910
                                    0.000
                                                  0.3460
0
                                           C#
                                                             -1.828
Major
                 0.737
                                           F#
                                                  0.1510
        137373
                                    0.000
                                                            -5.559
Minor
        170267
                 0.131
                                    0.000
                                                  0.1030
                                                           -13.879
Minor
        152427
                 0.326
                                    0.000
                                           C#
                                                  0.0985
                                                           -12.178
Major
         82625
                 0.225
                                    0.123
                                                  0.2020
                                                           -21.150
Major
   speechiness
                  tempo time signature
                                         valence
0
        0.0525
                166.969
                                    4/4
                                            0.814
1
        0.0868
                174.003
                                    4/4
                                            0.816
2
        0.0362
                 99.488
                                    5/4
                                            0.368
3
        0.0395
                171.758
                                    4/4
                                            0.227
        0.0456
                140.576
                                    4/4
                                            0.390
df raw.shape
(232725, 18)
```

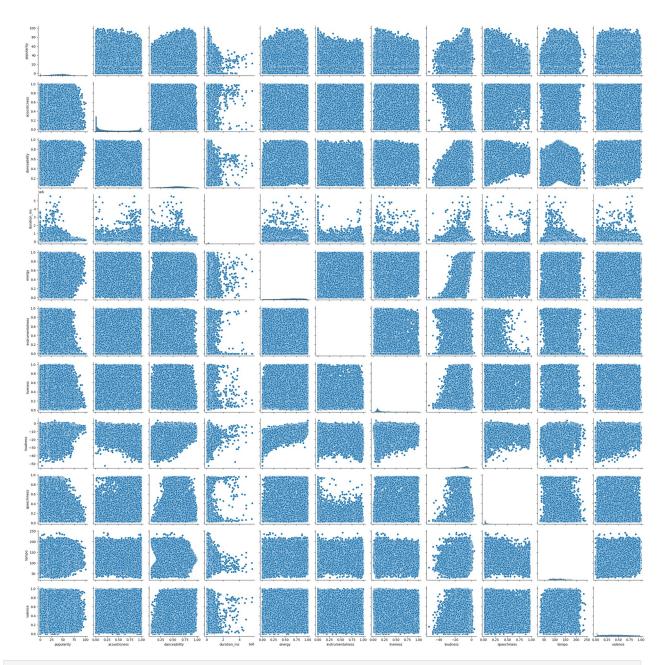
```
genre list = df raw['genre'].unique()
genre list
array(['Movie', 'R&B', 'A Capella', 'Alternative', 'Country', 'Dance',
       'Electronic', 'Anime', 'Folk', 'Blues', 'Opera', 'Hip-Hop', 
"Children's Music", 'Children's Music', 'Rap', 'Indie',
       'Classical', 'Pop', 'Reggae', 'Reggaeton', 'Jazz', 'Rock',
'Ska',
       'Comedy', 'Soul', 'Soundtrack', 'World'], dtype=object)
#Preprocessina
df = df raw[['genre', 'popularity',
'acousticness', 'danceability', 'duration ms', 'energy', 'instrumentalnes
     'key', 'liveness', 'loudness', 'mode', 'speechiness',
      'tempo','time signature','valence']]
df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 232725 entries, 0 to 232724
Data columns (total 15 columns):
#
     Column
                       Non-Null Count
                                         Dtvpe
     -----
- - -
0
                       232725 non-null object
     genre
     popularity
                       232725 non-null int64
1
 2
     acousticness
                       232725 non-null float64
 3
     danceability
                       232725 non-null float64
4
                       232725 non-null int64
     duration ms
 5
                       232725 non-null float64
     energy
    instrumentalness 232725 non-null float64
 6
 7
                       232725 non-null object
    kev
                       232725 non-null float64
 8
    liveness
 9
    loudness
                       232725 non-null float64
10 mode
                       232725 non-null object
 11 speechiness
                       232725 non-null float64
12 tempo
                       232725 non-null float64
                       232725 non-null
 13 time signature
                                         object
14 valence
                       232725 non-null float64
dtypes: float64(9), int64(2), object(4)
memory usage: 26.6+ MB
#Preprocessing
df['genre'] = df['genre'].str.replace("Children's Music", "Childrens
Music")
df['genre'].value counts()
C:\Users\mithu\AppData\Local\Temp\ipykernel 6260\121275794.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row indexer,col indexer] = value instead
```

```
See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['genre'] = df['genre'].str.replace("Children's Music", "Childrens
Music")
genre
                    9681
Comedy
Soundtrack
                    9646
Indie
                    9543
Jazz
                    9441
Pop
                    9386
                    9377
Electronic
Children's Music
                    9353
Folk
                    9299
                    9295
Hip-Hop
Rock
                    9272
Alternative
                    9263
Classical
                    9256
                    9232
Rap
World
                    9096
Soul
                    9089
Blues
                    9023
R&B
                    8992
                    8936
Anime
Reggaeton
                    8927
Ska
                    8874
Reggae
                    8771
Dance
                    8701
Country
                    8664
Opera
                    8280
Movie
                    7806
Childrens Music
                    5403
A Capella
                     119
Name: count, dtype: int64
df['genre'] = df['genre'].str.replace("Children's Music", "Childrens
Music")
df['genre'].value counts()
C:\Users\mithu\AppData\Local\Temp\ipykernel 6260\497271231.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row indexer,col indexer] = value instead
See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['genre'] = df['genre'].str.replace("Children's Music", "Childrens
Music")
```

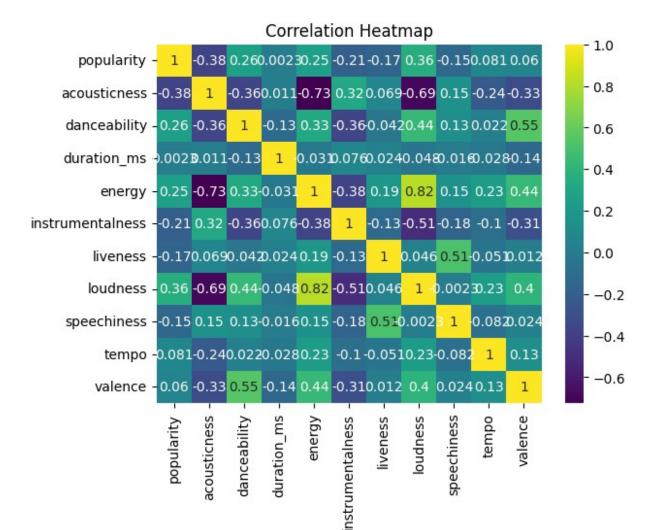
```
genre
Childrens Music
                   14756
Comedy
                    9681
Soundtrack
                    9646
Indie
                    9543
Jazz
                    9441
Pop
                    9386
Electronic
                    9377
                    9299
Folk
Hip-Hop
                    9295
Rock
                    9272
Alternative
                    9263
Classical
                    9256
                    9232
Rap
World
                    9096
Soul
                    9089
Blues
                    9023
R&B
                    8992
Anime
                    8936
                    8927
Reggaeton
                    8874
Ska
Reggae
                    8771
                    8701
Dance
Country
                    8664
                    8280
0pera
Movie
                    7806
A Capella
                     119
Name: count, dtype: int64
genre_counts = df['genre'].value_counts()
plt.figure(figsize=(15, 5))
genre_counts.plot(kind='bar', color='skyblue')
plt.title('Genre Distribution in Spotify Dataset')
plt.xlabel('Genre')
plt.ylabel('Count')
plt.xticks(rotation=25)
plt.tight layout()
for bar in plt.gca().containers[0]:
    plt.text(bar.get x() + bar.get width() / 2, bar.get height(),
int(bar.get_height()), ha='center', va='bottom')
plt.show()
```



```
#Checking out relationships of variables in this dataset
import seaborn as sns
sns.pairplot(df[['popularity',
'acousticness','danceability','duration_ms','energy','instrumentalness
', 'liveness', 'loudness', 'speechiness', 'tempo','valence']])
plt.show()
```



```
#Checking out the correlation matrix
correlation_matrix = df[['popularity',
   'acousticness', 'danceability', 'duration_ms', 'energy', 'instrumentalnes
s', 'liveness', 'loudness', 'speechiness', 'tempo', 'valence']].corr()
sns.heatmap(correlation_matrix, annot=True, cmap='viridis')
plt.title('Correlation Heatmap')
plt.show()
```



```
#Training a random forest classifier to predict genres
X= df[['popularity',
'acousticness', 'danceability', 'duration_ms', 'energy', 'instrumentalnes
s', 'liveness', 'loudness', 'speechiness', 'tempo', 'valence']].values
y = df['genre'].values
from sklearn.ensemble import RandomForestClassifier
from sklearn.model selection import train test split
from sklearn.metrics import accuracy score, precision score,
recall score, fl score, confusion matrix
X train, X test, y train, y test = train test split(X, y,
test size=\frac{0.2}{100}, random state=\frac{42}{100}
rf classifier = RandomForestClassifier(n estimators=100,
random state=42)
rf classifier.fit(X_train, y_train)
y pred = rf classifier.predict(X test)
cm rf = confusion matrix(y test, y pred)
```

```
plt.figure(figsize=(8, 6))
sns.heatmap(cm_rf, annot=True, fmt='g', cmap='Blues', cbar=False)
plt.xlabel('Predicted Genre')
plt.ylabel('True Genre')
plt.title('Confusion Matrix')
plt.show()
```

Confusion Matrix

```
0-3 0 0 1 1 0 0 0 0 0 0 0 0 9 4 0 0 0 0
                                                          1 0 0 0 0
 - 0 224 12 45 544 3
                    2 128 65 57 67 81 91 26 0 0 25 142 32 18 24 134 13 53 4
 - 0 51 04865 20 42 0 29 2 47 2
                                  0 1 10 49 61 1
                                                   0 0 13 40 7 148 6 135 71
 - 0 55 61 728 14 19 0 70 2 77 72 0 7 128 17 6
                                                      0 91 49 60 123 94 17 89
                                               1 1
 - 0 464 8 14 878 17 5 69 109 12 71 117215 14 161 4 67 162 91 16
  - 0 28 37 31 15 13 0 1
                          0 20 16 0
                                    4 44 29 160 0
                                                   1
                                                      0
                                                        0
                                                              2 5
                                                           1
  - 0 0 3 15 16 0 82 1 0 0 1 3 1 4 51 5
                                               0
                                                  1
                                                     1 7
                                                           8 0 10 5
  - 0 75 22 52 46 0 3 658 40 14 186 15 24 29 8
                                             0 75 33 19 35 53 220 22 62 0
_{\infty} - 0 51 1 3 106 0 0 69 143 47 41 103112 4 8 0 412239 98 9 26 131 7 51 0
o - 0 47 50 64 41 1 2 17 60 987 10 18 12 183 3 0 10 15 2 55 34 19 28 37 58 88
 - 0 63 1 58 54 7
                    0 217 41 17 372 6 437105 2 3 71 33 0 9 6 207 18 82 16 29
          0 76 0 0 5 61 7 9 259 37 14 0 0 221123857 15 80 14 1 46
                    0 44 113 13 470 53 81 33 0 0 136177 58 7
         1 3 199 1
                                                           9 272 3 130 9
  - 0 17 9 161 18 86 3 37
                          4 229120 15 19 656 21 2 3 38
                                                     5 52 52 10 8 195 55 72
      0 43 45 185 52 14
                      8
                          2
                            0
                              21 1 4 17 829156 2
                                                      1 9 11 1 20
                                  0 0 6 60 42 0
              4 115 3
                              2
                                                   0 0
                       0
                          0
                            0
                                                        0 0
                                                              0
           0 44 2
                    0 40 371 6 59 238157 4 2 0 233132335 3 32 238 1 29 2 1
                    1 47 214 17 45 152151 41 0 0 136157126 41 44 11 2 359 0 12
  - 0 12 0 0 97 0
                    0 27 94 4 6 313 43 6 0 0 38611480 1 3 62 0 16
  - 0 25 32 104 23 0
                    1 50 21 59 14 47 11 39 14 0 9 37 7 706265 54 177 41 2 26
  - 0 6 38 30 6
                 0 2 41 30 50 5 127 3 12 5
                                             0 45 44 27 166 11 11 29 19 0 12
  - 0 66 3 31 323 2 2 180 84 5 225 14 250 8 0 0 261 7 53 26 4 238 9 14 7 22
  - 0 45 129111 19 1 6 35
                          9 34 16 4 5 22 24 0 2 2
                                                      0 22471 18 049 6
  - 0 35 3 110 40 1 1 113 48 63 143 68 144189 5
                                            0 40 355 15 72 51 55 5 183 15 24
  - 0 1 51 18 4 238 2 3
                          0 20 12 0 5 38 71 13
                                               0
                                                   1
                                                      0
                                                        0 0 1 2 2 37 39
   0 33 53 121 46 110 1
                      70
                         4 112 48
                                     7 15830 13
                                  1
                                                0
                                                   5
                                                      0
                                                       34 18 28 15 15 137776
                       7
                         8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
                                Predicted Genre
```

```
accuracy_rf = accuracy_score(y_test, y_pred)
accuracy_rf
0.36871844451605973
```

Closing thoughts

Due to the number of features as well as number of classes (genres) in this data, the model is struggling to find patterns and achieve a good accuracy. We can experiment with neural networks in the future for better performance.

Project by Mithun Meenakshi S.