SC1015 Mini-Project

Agarwal Ananya
Banerjee Mohor
Goenka Shrivardhan
(Team 6)

HOME

TABLE OF CONTENTS

PLAYLIST

Problem Statement

We want to predict what influences the popularity index of a song more - Lyrics or Parameters like its Danceability, Energy, Tempo etc.

Aim of the Problem

We want the artists to know what feature of the song they should focus on more so that they can release more chartbusters.

Mars Is a Cold Place
The 15th Planet

2:54

3:49

Dataset 1 : Source - Kaggle

https://www.kaggle.com/datasets/lehaknarnauli/spotify-datasets?select=tracks.csv

```
In [4]: stats_data.head()
```

Out[4]:

	id	name	popularity	duration_ms	explicit	artists	id_artists	release_date	danceability	energy	key
0	35iwgR4jXetI318WEWsa1Q	Carve	6	126903	0	['Uli']	['45tIt06XoI0Io4LBEVpls']	1922-02-22	0.645	0.4450	0
1	021ht4sdgPcrDgSk7JTbKY	Capítulo 2.16 - Banquero Anarquista	0	98200	0	['Fernando Pessoa']	['14jtPCOoNZwquk5wd9DxrY']	1922-06-01	0.695	0.2630	0
2	07A5yehtSnoedViJAZkNnc	Vivo para Quererte - Remasterizado	0	181640	0	['Ignacio Corsini']	['5LiOoJbxVSAMkBS2fUm3X2']	1922-03-21	0.434	0.1770	1
3	08FmqUhxtyLTn6pAh6bk45	El Prisionero - Remasterizado	0	176907	0	['Ignacio Corsini']	['5LiOoJbxVSAMkBS2fUm3X2']	1922-03-21	0.321	0.0946	7
4	08y9GfoqCWfOGsKdwojr5e	Lady of the Evening	0	163080	0	['Dick Haymes']	['3BiJGZsyX9sJchTqcSA7Su']	1922	0.402	0.1580	3

(586672 X 20)

Dataset 2 : Source - Kaggle

https://www.kaggle.com/datasets/neisse/scrapped-lyrics-from-6-genres?select=lyrics-data.csv

```
In [14]: lyrics.head()
```

Out[14]:

	ALink	SName	SLink	Lyric	language
0	/ivete-sangalo/	Arerê	/ivete-sangalo/arere.html	Tudo o que eu quero nessa vida,\nToda vida, é\...	pt
1	/ivete-sangalo/	Se Eu Não Te Amasse Tanto Assim	/ivete-sangalo/se-eu-nao-te-amasse-tanto-assim...	Meu coração\nSem direção\nVoando só por voar\n...	pt
2	/ivete-sangalo/	Céu da Boca	/ivete-sangalo/chupa-toda.html	É de babaixá!\nÉ de balacubaca!\nÉ de babaixá!...	pt
3	/ivete-sangalo/	Quando A Chuva Passar	/ivete-sangalo/quando-a-chuva-passar.html	Quando a chuva passar\n\nPra quê falar\nSe voc...	pt
4	/ivete-sangalo/	Sorte Grande	/ivete-sangalo/sorte-grande.html	A minha sorte grande foi você cair do céu\nMin...	pt

(379931X5)

Data Cleaning

- Dropping of Duplicate Songs

```
In [8]: stats_data.drop_duplicates(subset='name', inplace=True)
```

- Dropping of N/A values

```
In [9]: stats_data.dropna(inplace=True)
```

Data Cleaning

- Raw Data

```
In [5]: stats_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 586672 entries, 0 to 586671
Data columns (total 20 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   id                586672 non-null  object
 1   name              586601 non-null  object
 2   popularity        586672 non-null  int64
 3   duration_ms       586672 non-null  int64
 4   explicit          586672 non-null  int64
 5   artists           586672 non-null  object
 6   id_artists        586672 non-null  object
 7   release_date      586672 non-null  object
 8   danceability      586672 non-null  float64
 9   energy            586672 non-null  float64
 10  key               586672 non-null  int64
 11  loudness          586672 non-null  float64
 12  mode              586672 non-null  int64
 13  speechiness       586672 non-null  float64
 14  acousticness      586672 non-null  float64
 15  instrumentalness  586672 non-null  float64
 16  liveness          586672 non-null  float64
 17  valence           586672 non-null  float64
 18  tempo             586672 non-null  float64
 19  time_signature    586672 non-null  int64
dtypes: float64(9), int64(6), object(5)
memory usage: 89.5+ MB
```

- Cleaned Data

```
In [10]: stats_data.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 446474 entries, 0 to 586670
Data columns (total 20 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   id                446474 non-null  object
 1   name              446474 non-null  object
 2   popularity        446474 non-null  int64
 3   duration_ms       446474 non-null  int64
 4   explicit          446474 non-null  int64
 5   artists           446474 non-null  object
 6   id_artists        446474 non-null  object
 7   release_date      446474 non-null  object
 8   danceability      446474 non-null  float64
 9   energy            446474 non-null  float64
 10  key               446474 non-null  int64
 11  loudness          446474 non-null  float64
 12  mode              446474 non-null  int64
 13  speechiness       446474 non-null  float64
 14  acousticness      446474 non-null  float64
 15  instrumentalness  446474 non-null  float64
 16  liveness          446474 non-null  float64
 17  valence           446474 non-null  float64
 18  tempo             446474 non-null  float64
 19  time_signature    446474 non-null  int64
dtypes: float64(9), int64(6), object(5)
memory usage: 71.5+ MB
```

Data Cleaning

- Dropping of Duplicate Songs

```
In [17]: lyrics.drop_duplicates(subset='SName', inplace=True)
```

- Dropping of N/A values

```
In [16]: lyrics.dropna(inplace=True)
```

- Renaming of Sname to name

```
In [18]: #Renaming the column 'SName' to 'name' to match the column name in the stats_data dataframe
lyrics.rename(columns={'SName':'name'}, inplace=True)
```

Data Merging

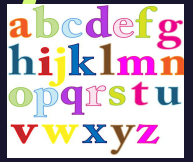
```
In [19]: merged = pd.merge(stats_data, lyrics, on='name')
```

```
In [21]: merged.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 41921 entries, 0 to 41920
Data columns (total 24 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   id                41921 non-null  object
 1   name              41921 non-null  object
 2   popularity        41921 non-null  int64
 3   duration_ms       41921 non-null  int64
 4   explicit          41921 non-null  int64
 5   artists           41921 non-null  object
 6   id_artists        41921 non-null  object
 7   release_date      41921 non-null  object
 8   danceability      41921 non-null  float64
 9   energy            41921 non-null  float64
 10  key               41921 non-null  int64
 11  loudness          41921 non-null  float64
 12  mode              41921 non-null  int64
 13  speechiness       41921 non-null  float64
 14  acousticness      41921 non-null  float64
 15  instrumentalness  41921 non-null  float64
 16  liveness          41921 non-null  float64
 17  valence           41921 non-null  float64
 18  tempo             41921 non-null  float64
 19  time_signature    41921 non-null  int64
 20  ALink             41921 non-null  object
 21  SLink             41921 non-null  object
 22  Lyric             41921 non-null  object
 23  language          41921 non-null  object
dtypes: float64(9), int64(6), object(9)
memory usage: 8.0+ MB
```

Shape :
(41921X23)

Data Preprocessing (NLP)



1) Conversion of characters to lowercase

- This has been done for consistency while processing.

2) Stopwords Removal :

- Removal of all non-alphabetic characters and whitespace
- Removed because they do not carry much meaning.

TextStrings
Apple)[){_+\.-/685)_
Orange1):>3._{26$5/_8
Pear'.@1$-+|8]92.3_
Watermelon~}][]6'\$#8~]\^
James]0=_8*4@$~]'"^3
Apple4,;|4?[-4!9&8?^
Pear)5`09*]=4#^+_{9
Orange%^-)*_|7<\~]|(8
Peach~5{80"'1&<[(>[?
Melon+@64625.]307[)7
Cherry08#5!)7=,[[=#>2

Data Preprocessing (NLP)

3) Stemming:

- Process of reducing a word to its base form by removing its affixes.
- Normalizes words and improves natural language processing tasks.

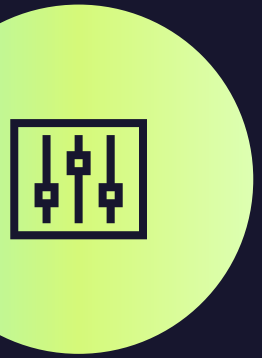
4) Tokenization:

- Process of breaking down a text into individual words or phrases, known as tokens.
- Facilitates NLP tasks like text analysis.

HOME

TABLE OF CONTENTS

PLAYLIST

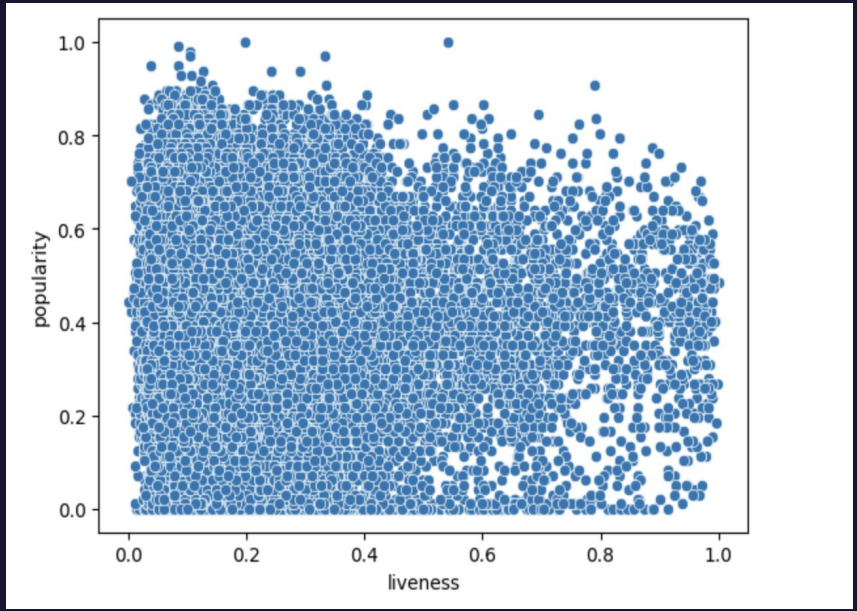


Exploratory Data Analysis

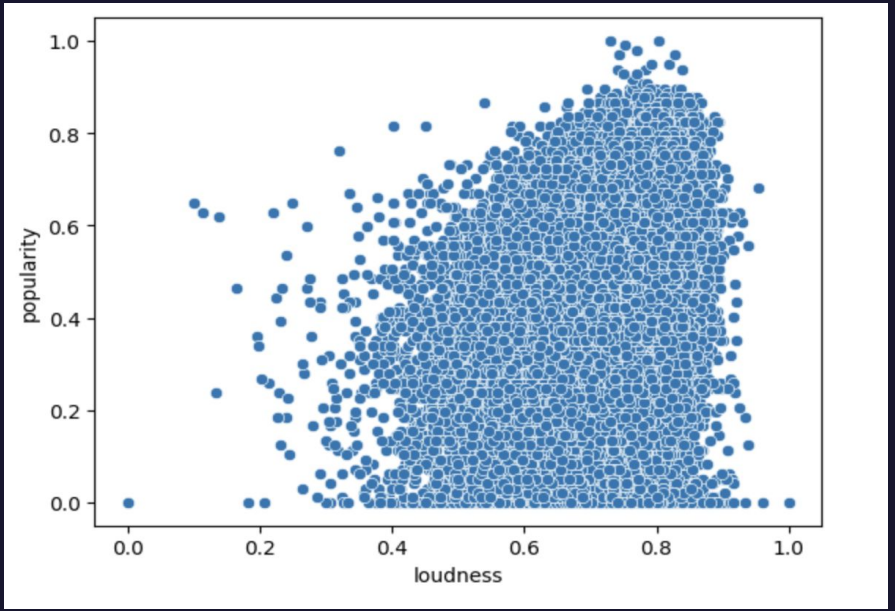
Mars Is a Cold Place
The 15th Planet

2:54 3:49

Scatterplots to show correlation between popularity and other variables visually

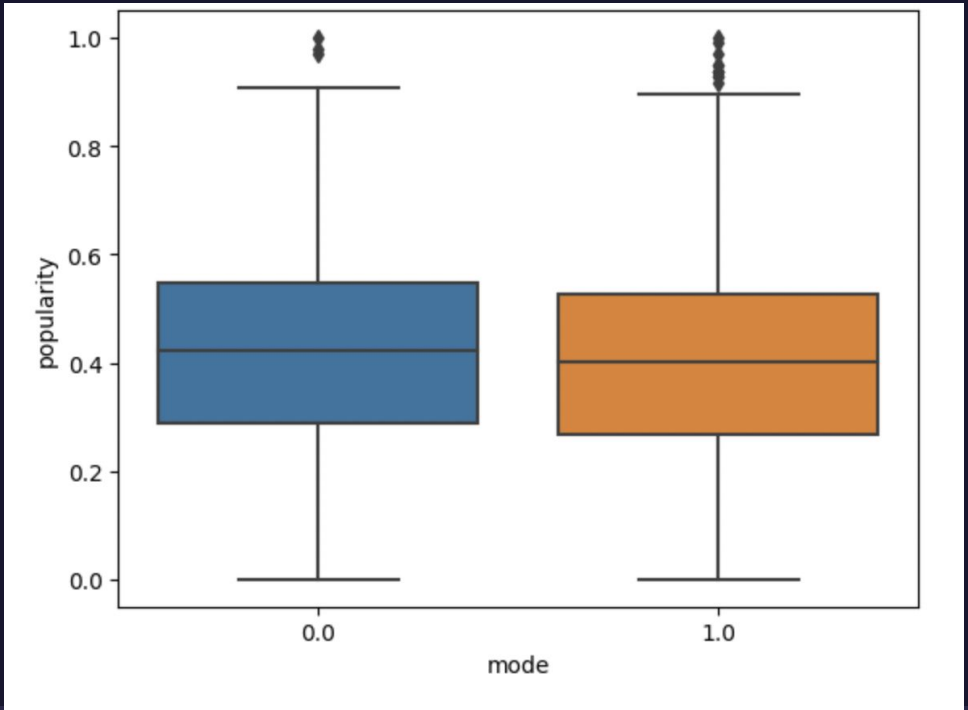


Very poor negative Correlation

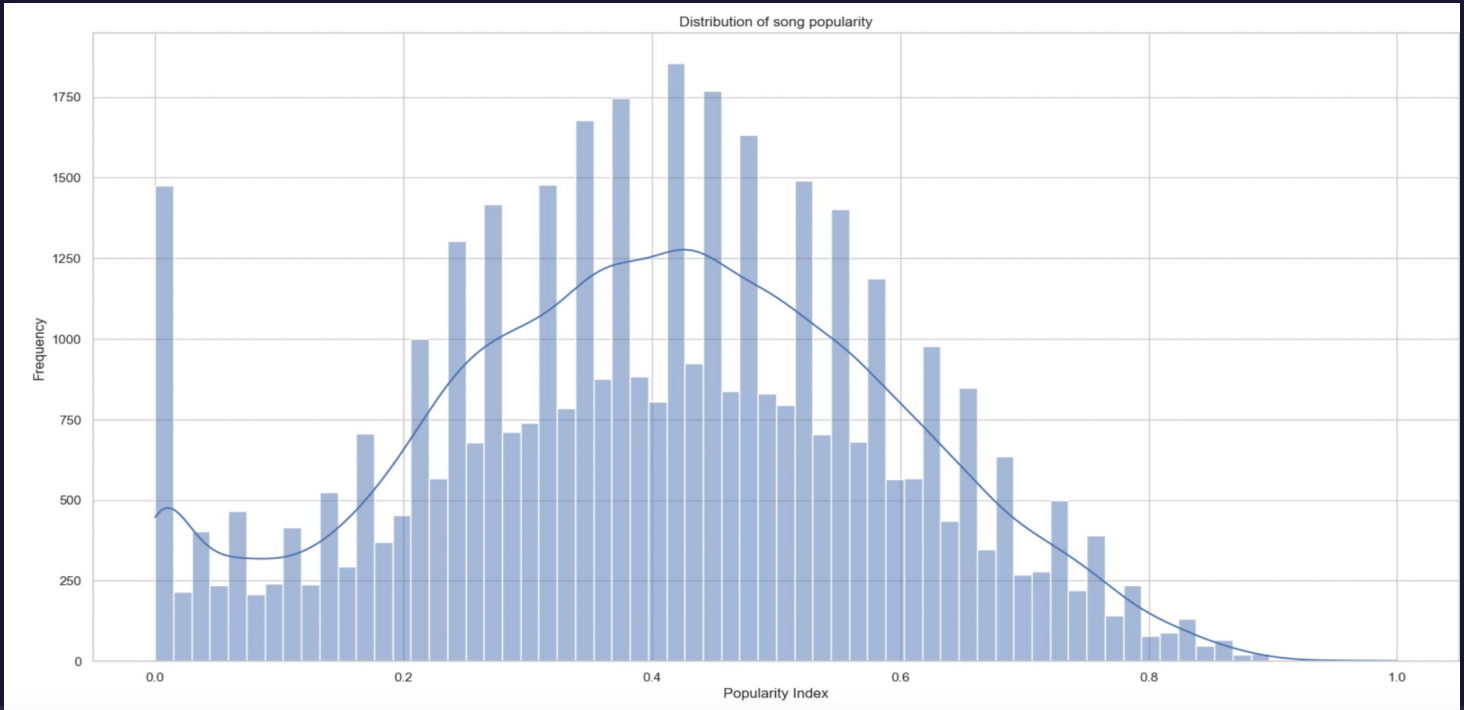


Fairly positive Correlation

Boxplot for popularity against categorical variables



KDE and Histogram of Popularity Index

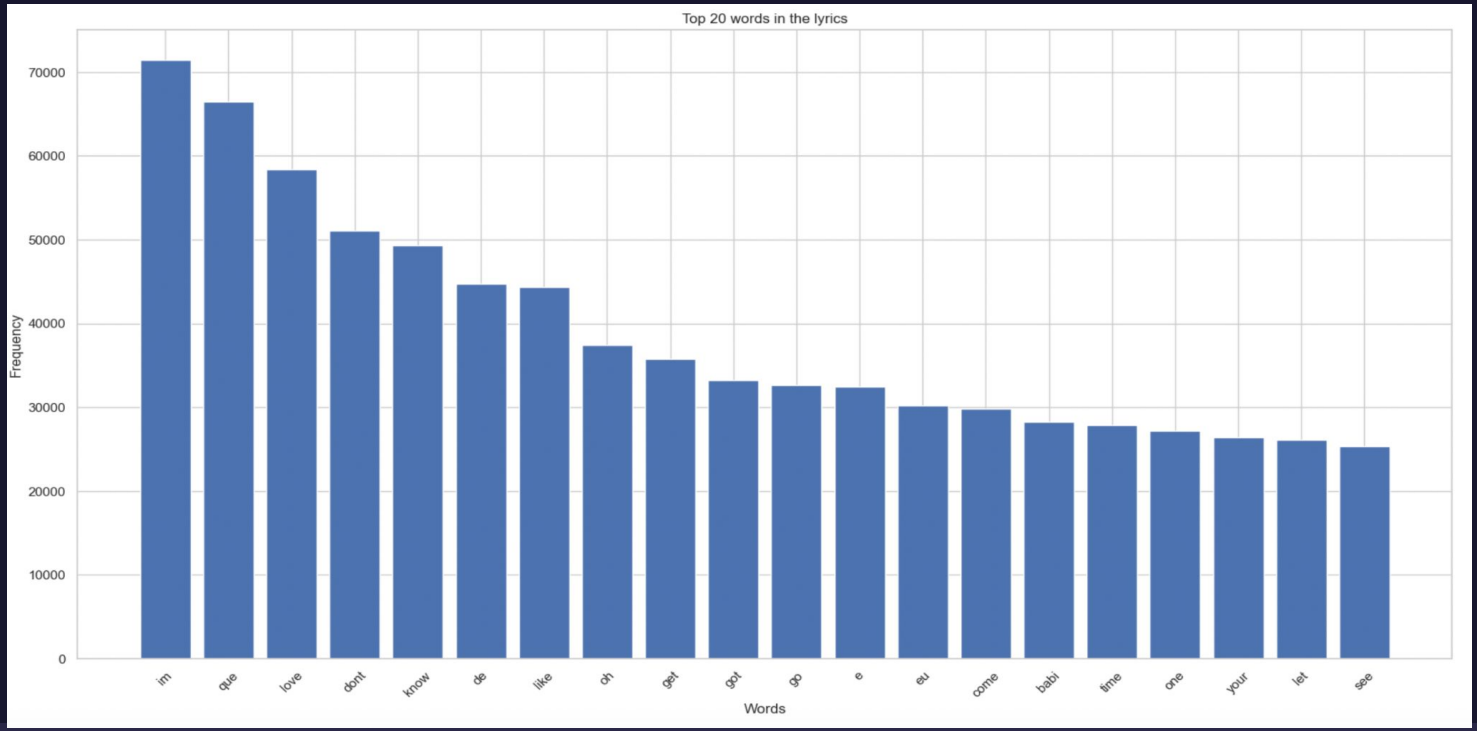


Mars Is a Cold Place
The 15th Planet

2:54

3:49

Bar plot showing Frequency of Top 20 words



Mars Is a Cold Place
The 15th Planet

2:54

3:49

Models

- Train-test data split 80:20

Sequential Neural Network

```
In [9]: Model1 = Sequential()

        Model1.add(Dense(32, activation='relu', input_dim=X_train.shape[1]))
        Model1.add(Dense(16, activation='relu'))
        Model1.add(Dense(8, activation='relu'))
        Model1.add(Dense(1, activation='linear'))
```

Layer	1st	2nd	3rd	4th
Node	32 with ReLU	16 with ReLU	8 with ReLU	1 with Linear Activation Function

- Supervised Learning Approach

Random Forest

```
In [17]: regressor = RandomForestRegressor(n_estimators = 100)
         regressor.fit(X_train, y_train)

Out[17]: RandomForestRegressor()
```

- Trained on randomly selected subset of the training data
- N_estimators specifies the number of decision trees to include in random forest

Linear Regression

```
In [16]: # Create the regression model
         model = LinearRegression()
```

- Finds best-fitting straight line relationship between feature and target
- One of the simplest models
- Supervised Learning Approach

Models

- Train-test data split 80:20

LSTM Model

```
# Build the model
embedding_dim = 100
model = Sequential()
model.add(Embedding(vocab_size, embedding_dim, input_length=max_len))
model.add(Bidirectional(LSTM(64, return_sequences=True)))
model.add(Bidirectional(LSTM(32)))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='linear'))
```

Layer	1st	2nd	3rd	4th	5th	6th
Node	Embedding layer	LSTM layer with 64 units	LSTM layer with 32 units	64 units with ReLu	Dropout layer	1 unit Linear Activation

Linear Regression

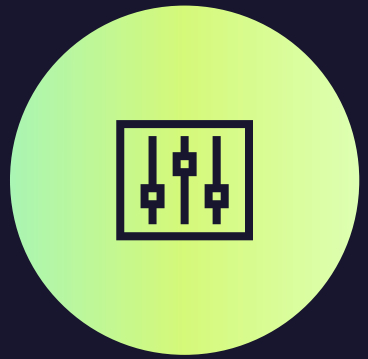
```
In [14]: # Train the model
         model = LinearRegression()
         model.fit(X_train_tf, y_train)
```

- Finds best-fitting straight line relationship between lyrics and target.

HOME

TABLE OF CONTENTS

PLAYLIST



Results

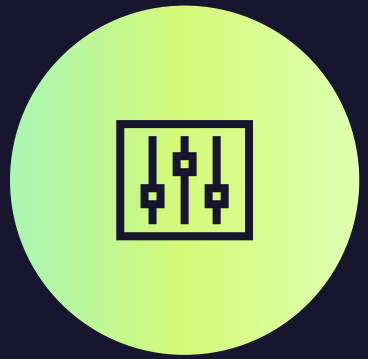
Mars Is a Cold Place
The 15th Planet

2:54 3:49

HOME

TABLE OF CONTENTS

PLAYLIST



1) Statistical Data

Mars Is a Cold Place
The 15th Planet

2:54 3:49

Neural Network

```
In [14]: mean_absolute_error(Y_test, predictions)
Out[14]: 0.1270828745284773

In [15]: mean_squared_error(Y_test, predictions)
Out[15]: 0.026014470953904592
```

Linear Regression

```
# Evaluate the model's performance
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("Mean Squared Error: ", mse)
print("R^2 Score: ", r2)
```

```
Mean Squared Error:  0.027815844346643 24
R^2 Score:  0.22851708747705524
```

Random Forest

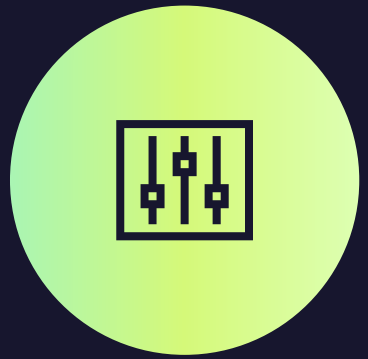
```
In [11]: mean_absolute_error(y_test, y_pred)

Out[11]: 0.1257995799240585

In [12]: np.sqrt(mean_squared_error(y_test, y_pred))

Out[12]: 0.1596036211433357
```

"OVERFITTING IN RANDOM FOREST MODEL"

HOME

TABLE OF CONTENTS

PLAYLIST



2) Lyrics

Mars Is a Cold Place
The 15th Planet

2:54 3:49

LSTM

```
In [14]: # Calculate Mean Absolute Error
         mae = mean_absolute_error(y_test, y_pred)
         print(f'Mean Absolute Error: {mae}')

         Mean Absolute Error: 0.16926953782466247

In [16]: #Calculate Mean Squared Error
         mse = mean_squared_error(y_test, y_pred)
         print("Mean Squared Error:", mse)

         Mean Squared Error: 0.04508657011284038
```

Linear Regression

```
In [14]: mean_squared_error(predictions, y_test)

Out[14]: 10.760397233843237
```

HOME

TABLE OF CONTENTS

PLAYLIST

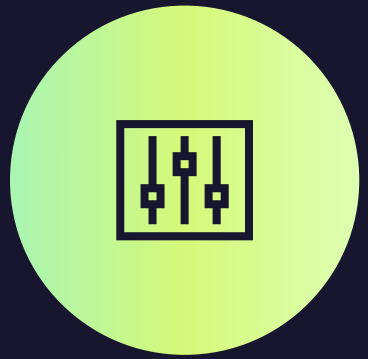
01 *Problem Statement*

02 *Data Extraction and Cleaning*

03 *EDA & Pre-Processing*

04 *Models*

05 *Results*



Inference

Mars Is a Cold Place
The 15th Planet

2:54 3:49

Final Results

```
In [15]: mean_squared_error(Y_test, predictions)
Out[15]: 0.026014470953904592
```

Statistical data

```
In [16]: #Calculate Mean Squared Error
         mse = mean_squared_error(y_test, y_pred)
         print("Mean Squared Error:", mse)

         Mean Squared Error: 0.04508657011284038
```

Lyrics

HOME

TABLE OF CONTENTS

PLAYLIST

Thanks!

Mars Is a Cold Place
The 15th Planet

2:54 3:49