Lyric Generation

By: Katarina Salcedo

Business Problem

Motivation

- On average it takes anywhere from 3-30 months to create albums
- Assist the Red Hot Chili Peppers in creating new hit songs for their next album
    - Announced the release of a new LP in 2019

Goal

- Provide an algorithm that can help write new songs
    - Predict the popularity of the given song/lyrics

Data

- Data includes 118 songs from all 10 of RHCP's albums
- Song lyrics were collected using lyricsgenius
- Audio features (danceability, energy, key, loudness, speechiness, etc) along with popularity were collected using Spotipy

Data Cleaning

Lyrics:

- Tokenize words
- Removal of stop words
  - Total tokens = 17177
  - Unique tokens = 3606
- Converted words to matrix (each unique token is represented by numbers/feature index)

Audio Features:

- Dropped unnecessary columns (uri, track, track_number, album)

EDA - Word Clouds

Most common words

Most common bigrams

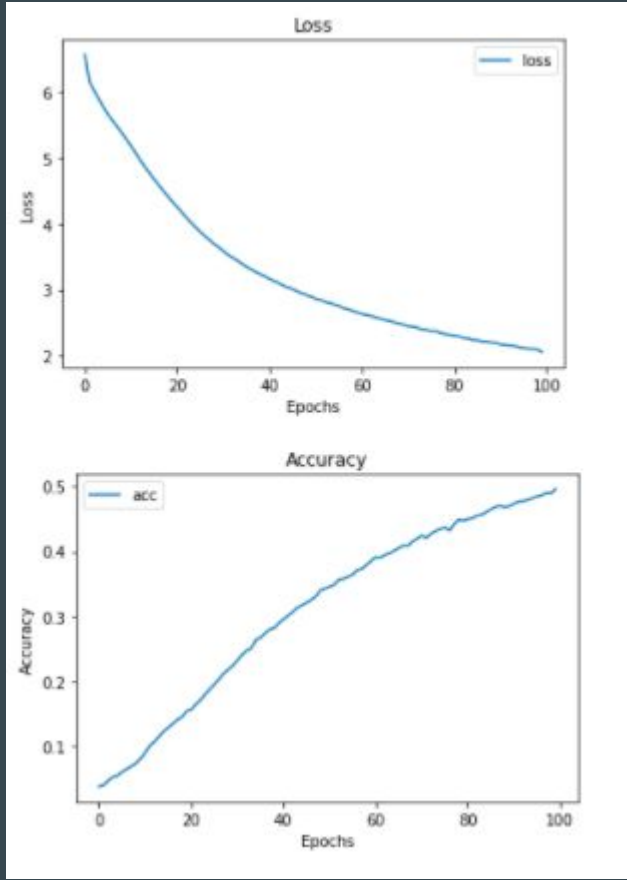
EDA - Popularity

Most popular songs (score=73):

1. Funky Crime
2. If You Have to Ask
3. Dark Necessities
4. Snow
5. Aeroplane

LSTM Network

- Long short-term memory network: a type of Recurrent Neural Network (RNN) that has feedback connections - capable of learning order dependence in sequence prediction
- Model architecture:
  - Embedding, LSTM, Dropout, Dense Layers
  - Total parameters = 409,213
- Metrics:
  - Accuracy = 0.4954
  - Loss = 2.05



Lyrics

you can kick my face for pop

but i can't contain me i

ain't up i am okay snow on the

mountain wasting away in your skin

we are the ones that will make

the world i really find

beautiful around i'll wait for you

baby oughta celebrate you can't torture me please

don't forsake me i'm contagious than fine

okay its stated sweep solitaire

Regression - Audio Features

- Predict audio features from song lyrics - Linear Support Vector Regression

```
danceability regression metrics:
Train Root Mean Square Error: 1.0314003630661054e-05
Test Root Mean Square Error: 0.1488226224388052


energy regression metrics:
Train Root Mean Square Error: 1.2139023676228211e-05
Test Root Mean Square Error: 0.18805055175535762


key regression metrics:
Train Root Mean Square Error: 2.8667839714296797
Test Root Mean Square Error: 3.660084089254169


loudness regression metrics:
Train Root Mean Square Error: 2.8018838253358602
Test Root Mean Square Error: 3.1466453600914828


speechiness regression metrics:
Train Root Mean Square Error: 6.44216812859672e-06
Test Root Mean Square Error: 0.05569830468137379
```

```
acousticness regression metrics:
Train Root Mean Square Error: 6.374996936550328e-06
Test Root Mean Square Error: 0.2249109340461646


instrumentalness regression metrics:
Train Root Mean Square Error: 8.65448820475393e-06
Test Root Mean Square Error: 0.14933409163989655


liveness regression metrics:
Train Root Mean Square Error: 9.74269309865555e-06
Test Root Mean Square Error: 0.09126341133127915


valence regression metrics:
Train Root Mean Square Error: 1.5232217911343105e-05
Test Root Mean Square Error: 0.22379044468181605


tempo regression metrics:
Train Root Mean Square Error: 41.56028748642094
Test Root Mean Square Error: 46.40042250097185
```

Regression - Popularity

- Predicting song popularity from audio features
  - Base Model: Linear Support Vector Regression

```
Train Root Mean Square Error: 8.679211711870 89
Test Root Mean Square Error: 7.534477731702534

LinearSVR()
```

  - Final Model: Stochastic Gradient Descent Regressor

```
Train Root Mean Square Error: 6.912113913735776
Test Root Mean Square Error: 5.952769492933285

SGDRegressor(penalty='l1')
```

Conclusion

- **Results:**
  - Danceability : 0.448
  - Energy : 0.795
  - Key : 4.643
  - Loudness : -4.537
  - Speechiness : 0.074
  - Acousticness : 0.026
  - Instrumentalness : 0.039
  - Liveness : 0.139
  - Valence : 0.479
  - Tempo : 82.7
  - Popularity : 49.23
- Currently the model is able to generate a song with a predicted popularity score of 49.23.

- **Future Improvements:**
  - More data - based on genre?
  - Increase number of epochs in neural network
  - Add more layers to model
  - Tweak learning rate
  - Further reduce RMSE in popularity regression model

Thank You

...

Github