«Music is the answer»

Predicting song popularities on **Spotify**

Bern Winter School on Machine Learning 2020

Matthias Niggli - University of Basel

The general idea

University of Basel

Given a set of song characteristics, can neural networks distinguish very popular songs from other songs?

Is it even doable? Spoken differently, are the two categories even seperable based on observed characteristics?

Data

University of Basel

14’683 unique songs from **Spotify** dataset

https://www.kaggle.com/edalrami/19000-spotify-songs

Additional data on artists, albums and release dates gathered from the **Spotify**-API (using R package `spotifyR`).

https://developer.spotify.com/documentation/web-api/

https://www.rcharlie.com/spotifyr/

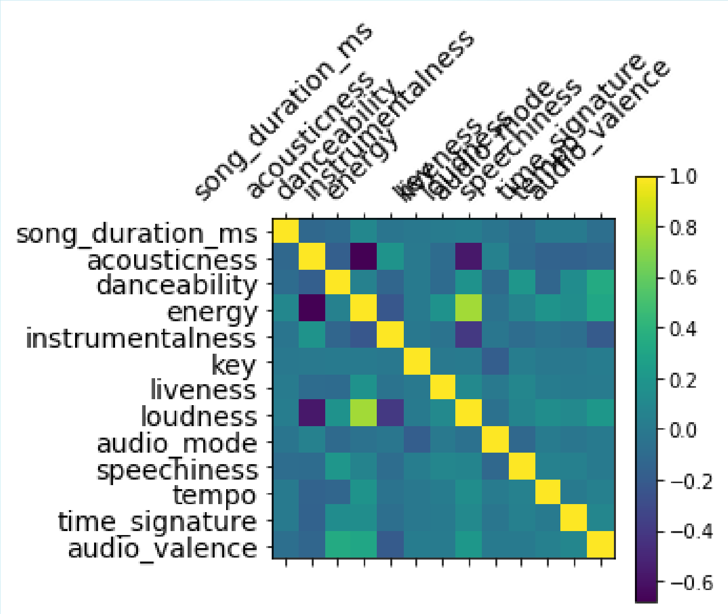
Definition: «Top-Songs» have a popularity score of at least 75.

A look at the data

University of Basel

14’ 683 samples

13 features



Number of songs: 14683															
	song_name	song_popularity	song_duration_	acousticness	danceability	energy	instrumentalness	key	liveness	loudness	audio_mode	speechiness	tempo	time_signature	audio_valence
0	Boulevard of Broken Dreams	73	262333	0.005520	0.496	0.682	0.000029	8	0.0589	-4.095	1	0.0294	167.060	4	0.474
1	In The End	66	216933	0.010300	0.542	0.853	0.000000	3	0.1080	-6.407	0	0.0498	105.256	4	0.370
2	Seven Nation Army	76	231733	0.008170	0.737	0.463	0.447000	0	0.2550	-7.828	1	0.0792	123.881	4	0.324
3	By The Way	74	216933	0.026400	0.451	0.970	0.003550	0	0.1020	-4.938	1	0.1070	122.444	4	0.198
4	How You Remind Me	56	223826	0.000954	0.447	0.766	0.000000	10	0.1130	-5.065	1	0.0313	172.011	4	0.574

First results

Model I (fully connected network):

```
model = tf.keras.models.Sequential([
        tf.keras.layers.Dense(30, activation='relu'),
        tf.keras.layers.Dense(10, activation='relu'),
        tf.keras.layers.Dense(2, activation='softmax'))
model.compile(optimizer='adam',
        loss='sparse_categorical_crossentropy',
        metrics=['accuracy'])
hist =  model.fit(x = x_train, y = y_train,
        epochs = 25, batch_size = 64,
        validation_data=[x_test, y_test])
```

➢ wow incredible, validation accuracy is 93 % …

Balancing issues

University of Basel

But: Prediction on most popular songs in the test set

```
[False, False, False, False, False, False, False, False, False, False]
```

Sample is very unbalanced w.r.t. top-songs:

```
0    13737
1      946
```

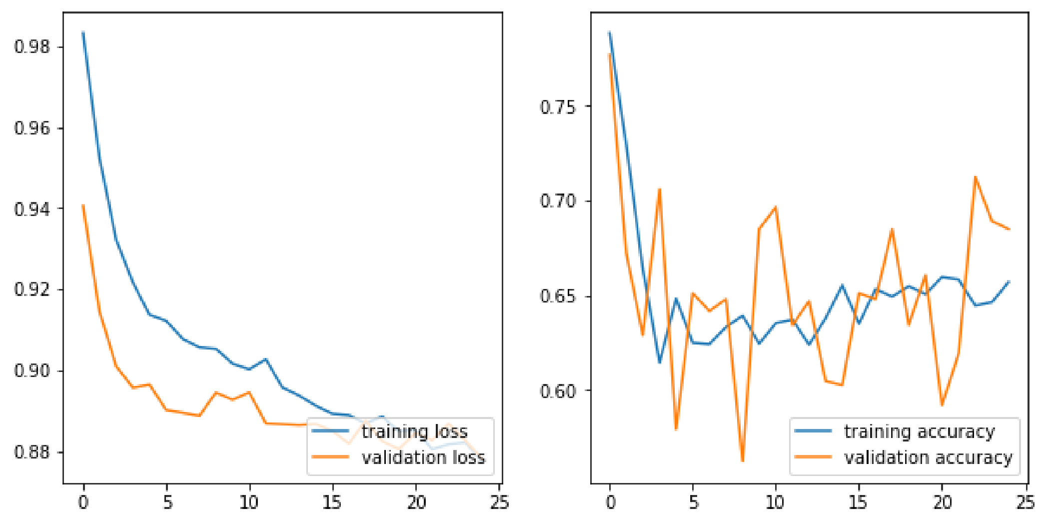
➢ Adjust the sample and the loss function to give top-songs more weight. Combine two approaches:

1. Downsample non-popular songs: at most `ratio` of 4 between classes
2. Weighted loss function

Balancing issues

Model II (fully connected network):

1. Overall validation accuracy: 69 %
2. Valdiation accuracy among
   top-songs (precision): 61%

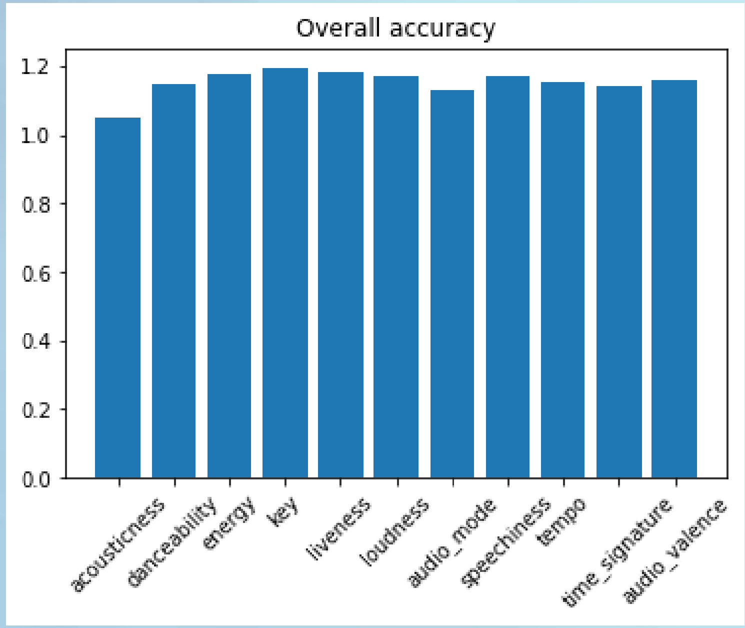


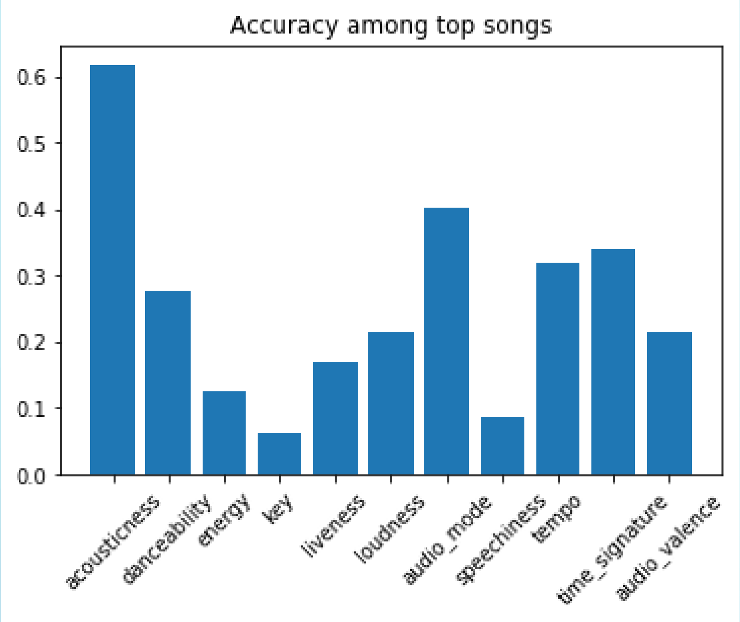
But: False positives among non-popular songs: 30%

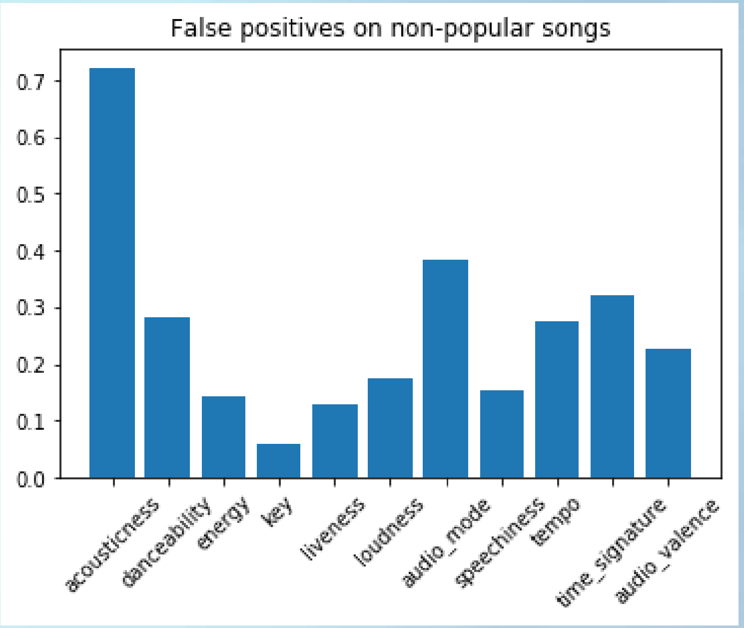
- ➢Network now mispredicts to many non-popular songs as top-songs.
- ➢Trade-off between accuracy among top-Songs and false positive among non-top Songs.

Feature importance

Evaluate by using **permutation importance** to get a rough estimate.
https://towardsdatascience.com/feature-importance-with-neural-network-346eb6205743







Illustrate for newly released songs

University of Basel

Pearl Jam
Gigaton
["Who Ever Said"]

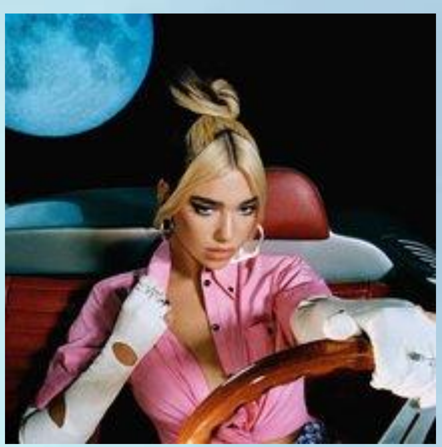
Top-Song: 20 %
No Top-Song: 80 %

Nine Inch Nails
Ghosts V: Together
"Letting Go While Holding On"

Top-Song: 1 %
No Top-Song: 99 %

The Weekend
After Hours
"Alone Again"

Top-Song: 11 %
No Top-Song: 89 %



Dua Lipa
Future Nostalgia
"Future Nostalgia"

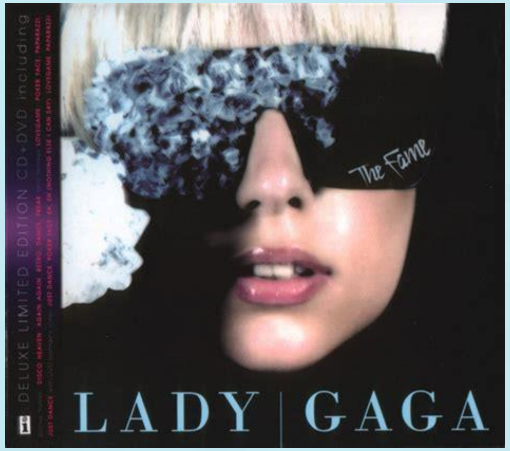
Top-Song: 45 %
No Top-Song: 55 %

https://www.spotifynewmusic.com/index.php?sort=rev&date=

Illustrations continued

Katy Perry
“Firework”

Top-Song: **53 %**
No Top-Song: 47 %



Lady Gaga
“Poker Face”

Top-Song: 43 %
No Top-Song: 57 %

The Rolling Stones
“Angie”

Top-Song: 17 %
No Top-Song: 83 %

Extensions: Time dimension

Taste for more recent songs might be different from older songs

- Subset the data to songs after 2010 and re-train the model:

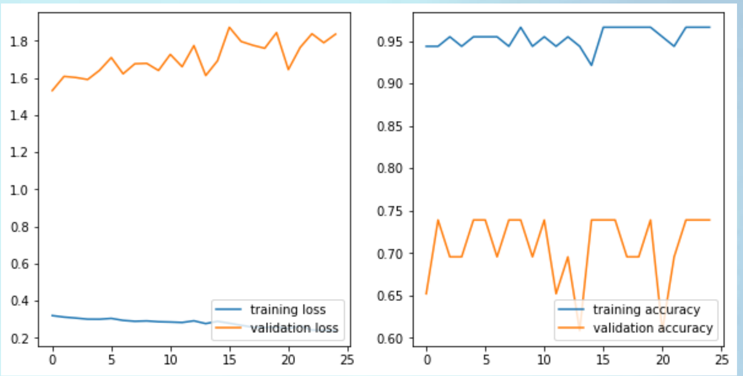
Model III (fully connected network):

  - Overall validation accuracy: 70 %
  - Valdiation accuracy top-songs (precision): 75 %
  - False Positives non-popular songs (FPR): 30 %

Extensions: Time dimension

Popularity of a song could be dependent on the success of previous songs of the same artists.

- add characteristics of past songs as features and re-train the model.



Model IV (fully connected network):

- 41 features now
- Overall evaluation accuracy: 74 %
- Valdiation accuracy top-songs (precision): 71 %
- False Positives non-popular songs (FPR): 25 %
- **But**: Only very limited number of samples. No meaningful results anymore

Extensions: Time dimension

University of Basel

Instead of adding characteristics of previous songs as features, train a Recurrent Neural Net with LSTM layer that directly includes past songs.

Model V (LSTM recurrent neural network):

```
model = tf.keras.models.Sequential()

model.add(tf.keras.layers.LSTM(10,

                         input_shape=(x_train.shape[1], x_train.shape[2]),

                         return_sequences = True))

model.add(tf.keras.layers.Dense(2, activation="softmax"))

model.compile(optimizer="adam", loss='sparse_categorical_crossentropy’,
              metrics=['accuracy’])
```

- Evaluation accuracy: 88 %
- **But**: very few sampls (not a meaningful application here).

Further steps & take-away’s

University of Basel

- Require more data or create additional samples (e.g. by using nearest neighbours)
- Subsample (e.g. only predict for Hip Hop, Metal etc.)
- Could also include lyrics

In general

- Seems to be difficult to clearly separate songs based on this data.
- Unbalanced data is a problem

Discussion