

# **Predict Song Skips on Spotify based on Acoustic Data**

**Sandesh Paudel**

**[Pdlsandesh317@gmail.com](mailto:Pdlsandesh317@gmail.com)**

**Technocolab Software's**

**2021-11-16**

## Overview:

Spotify, a music streaming company, have the problem of incorporating the sequential nature of a listening session to predict whether a user will skip a given music track or not. Modelling sequential music skips provides streaming companies like Spotify the ability to better understand the needs of the user base, resulting in a better user experience by reducing the need to manually skip certain music tracks. Given information about the first half of a Spotify user's listening session, the task is to predict whether each track in the second half of the session will be skipped or not. We model this task using a Gradient Boosting Tree.

The main task of this project is to predict which music tracks a user will skip in the second half of the session conditioned on the first half. This problem can be considered a type of session-based recommendation, since no explicit user profile is available, and the user preferences should therefore be estimated within the first half of the session. After preprocessing of data is completed we build the model using LightGBM and SVM to predict whether the user will skip the song or not

The accuracy of our LightGBM model is 87.5%. Our deployed code is released at.

[https://github.com/paudel-sandesh/Deployment\\_spotify\\_skip\\_predicton](https://github.com/paudel-sandesh/Deployment_spotify_skip_predicton)

## **Methodology:**

For this project, we follow simple machine learning life cycles all the steps in our life cycle are followed as below:

- 1: Datasets**
- 2: Data Preprocessing**
- 3: Data analysis and visualization**
- 4: Modelling**
- 5: Testing and result**
- 6: Deployment**

## **Datasets:**

Approximately 130 million listening sessions were provided for the training set, and each session contained between 10 and 20 tracks. The test set comprised approximately 30 million listening sessions, and detailed data was provided for the first half of each session, with the second half of each session to be predicted.

For this project, we only take a sample of 10,000 sessions data. where half of the tracks of each session are taken for the training and half of the tracks for testing.

Our datasets were divided into two main files where in the first file we have user behaviour features having 21 columns of the song like:

- Hour of the day
- Season length
- Context type
- History user behaviour

And the second file contains acoustics features having 29 columns like:

- Track beat strength
- Track acoustics vectors
- Tracks loudness etc.

## Data Preprocessing:

In order to pre-process the data for training, we merged user behaviour and acoustic features using track ids for each session. Later the following pre-processing techniques are employed

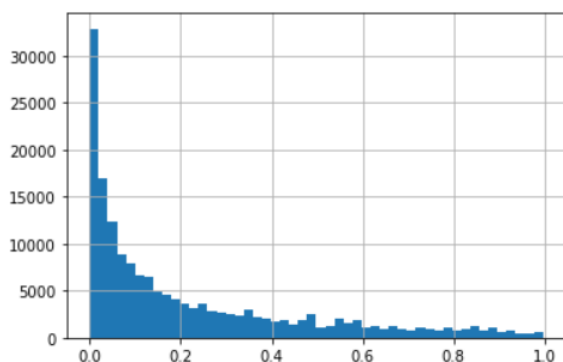
- 1, Handling missing values
- 2, Converting binary categorical features to numeric:
- 3, One hot Encoding
- 4, Outlier detection
- 5, Scaling using min-max scalar

## Data Analysis and Visualization:

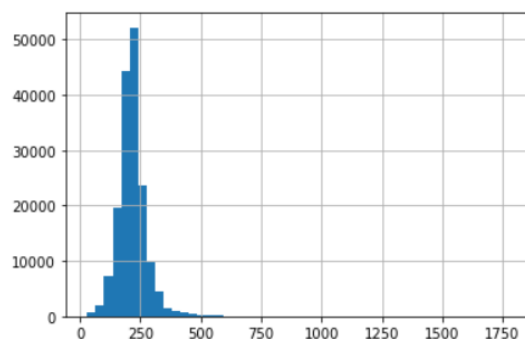
After preprocessing we gather useful information from the data. We find which features are more imported and how many values are there in these features this help a lot in data understanding and outlier detection

We did most of the data analysis using an external python library called **SweetViz** which help us to do advanced data analysis in an easy manner

Acousticness



Duration



21

## track\_id

VALUES: 167,880 (100%)  
MISSING: ---  
DISTINCT: 50,704 (30%)

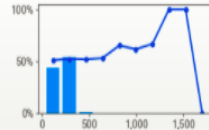
1,427 <1% L\_bacf06d3-9185-4183-84ea-f0db51475ce  
915 <1% L\_5718ab08-3a15-4d3f-9e63-42b2f6805e31  
785 <1% L\_8cd429b1-e0bf-464c-88f7-ac19240cbb80  
730 <1% L\_a66ea088-b357-449a-8a1e-64dd0b8dcb5  
719 <1% L\_77b02acb-1b1f-4b36-b8fc-2c3e01892b9a  
612 <1% L\_9a31436c-a57e-4d7a-9b0c-50f0deca33de  
600 <1% L\_0e3dec82-10b4-4911-8c2e-cd19249f7d2c  
162,092 97% (Other)

22

## duration

VALUES: 167,880 (100%)  
MISSING: ---  
DISTINCT: 33,549 (20%)  
ZEROS: ---

MAX 1,788 RANGE 1,758  
95% 311 IQR 57.3  
Q3 241 STD 60.2  
AVG 216 VAR 3,627  
MEDIAN 212  
Q1 184 KURT 23.9  
5% 131 SKEW 2.22  
MIN 30 SUM 36.2M

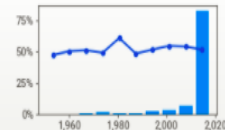


23

## release\_year

VALUES: 167,880 (100%)  
MISSING: ---  
DISTINCT: 69 (<1%)  
ZEROS: ---

MAX 2,018.0 RANGE 68.0  
95% 2,018.0 IQR 3.00  
Q3 2,018.0 STD 9.74  
MEDIAN 2,017.0 VAR 94.9  
AVG 2,013.4  
Q1 2,015.0 KURT 11.0  
5% 1,993.0 SKEW -3.24  
MIN 1,950.0 SUM 338.0M

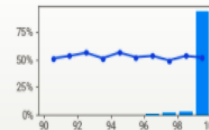


24

## us\_popularity\_estimate

VALUES: 167,880 (100%)  
MISSING: ---  
DISTINCT: 50,704 (30%)  
ZEROS: ---

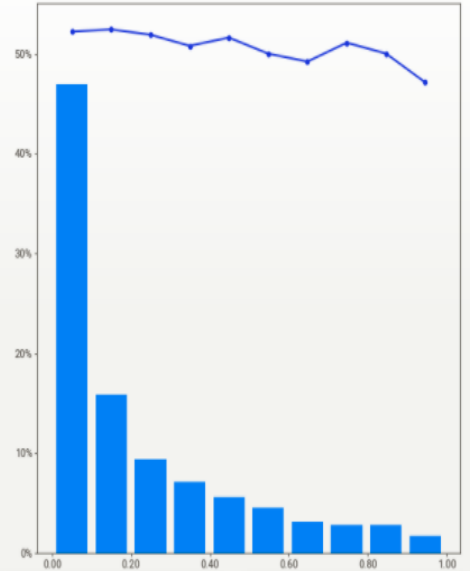
MAX 100.00 RANGE 9.98  
95% 100.00 IQR 0.085  
Q3 100.00 STD 0.893  
MEDIAN 99.99 VAR 0.797  
AVG 99.74  
Q1 99.91 KURT 46.4  
5% 98.68 SKEW -6.24  
MIN 90.02 SUM 16.7M



## acousticness

MISSING: ---

Auto 5 15 30



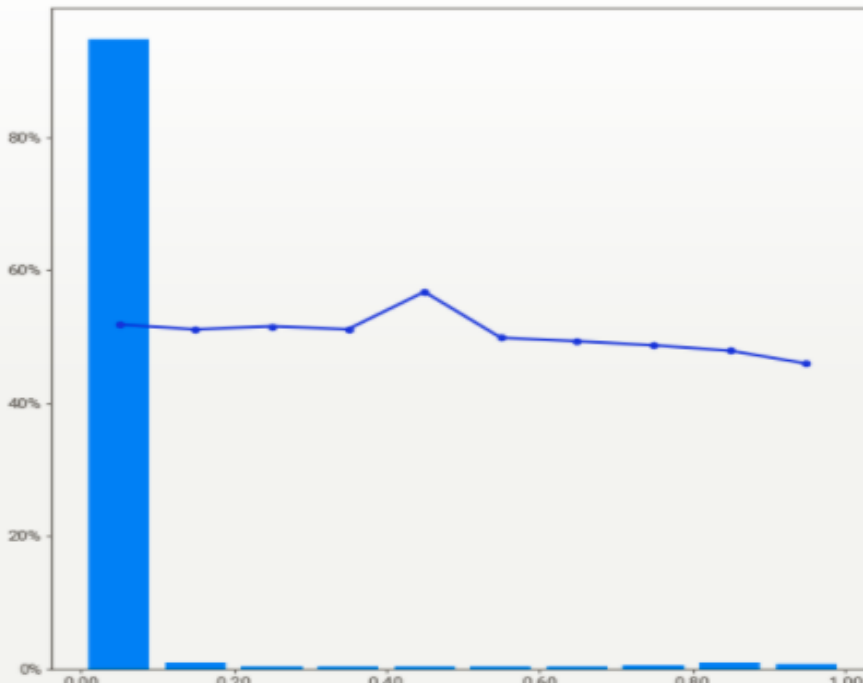
MOST FREQUENT VALUES

SMALLEST VALUES

## instrumentalness

MISSING: ---

Auto 5 15 30



## NUMERICAL ASSOCIATIONS

(PEARSON, -1 to 1)

acoustic_vector_1	-0.41
acoustic_vector_2	-0.34
acoustic_vector_0	0.33
loudness	-0.32
danceability	-0.20
bounciness	-0.17
best_strength	-0.15
dyn_range_mean	-0.14
acoustic_vector_7	0.14
acousticness	0.13
acoustic_vector_6	0.13
speechiness	-0.12
organism	0.11
energy	-0.11

## CATEGORICAL ASSOCIATIONS

(CORRELATION RATIO, 0 to 1)

time_signature	0.08
context_type	0.06
premium	0.04
hist_user_behavior_reason_st...	0.03
hist_user_behavior_is_shuffle	0.02
hist_user_behavior_reason_e...	0.02
date	0.02
skip_1	0.02
not_skipped	0.01
skip_3	0.01
skip_2	0.01
long_pause_before_play	0.01
no_pause_before_play	0.01
context_switch	0.01

## Modelling:

Two types of models were attempted:

- gradient boosted trees
- SVM

### Gradient Boosting trees:

A model was trained for each label, or target, track position. That is, a model was trained for the first track position to be predicted, another for the second track position to be predicted, and so forth. Identifying good parameters for training the model is a big task in model building. The parameters ultimately used in training the model were:

```
learning_rate':0.23,  
'boosting_type':'dart',  
'objective':'binary',  
'metric':['auc', 'binary_logloss'],  
'num_leaves':2500,  
'max_depth':5,  
"min_data_in_leaf": 200,  
"lambda_11": 15,  
"lambda_12": 70,  
"min_gain_to_split": 2,  
"bagging_fraction": 0.7,  
"bagging_freq": 1,  
"feature_fraction": 0.9
```

### SVM:

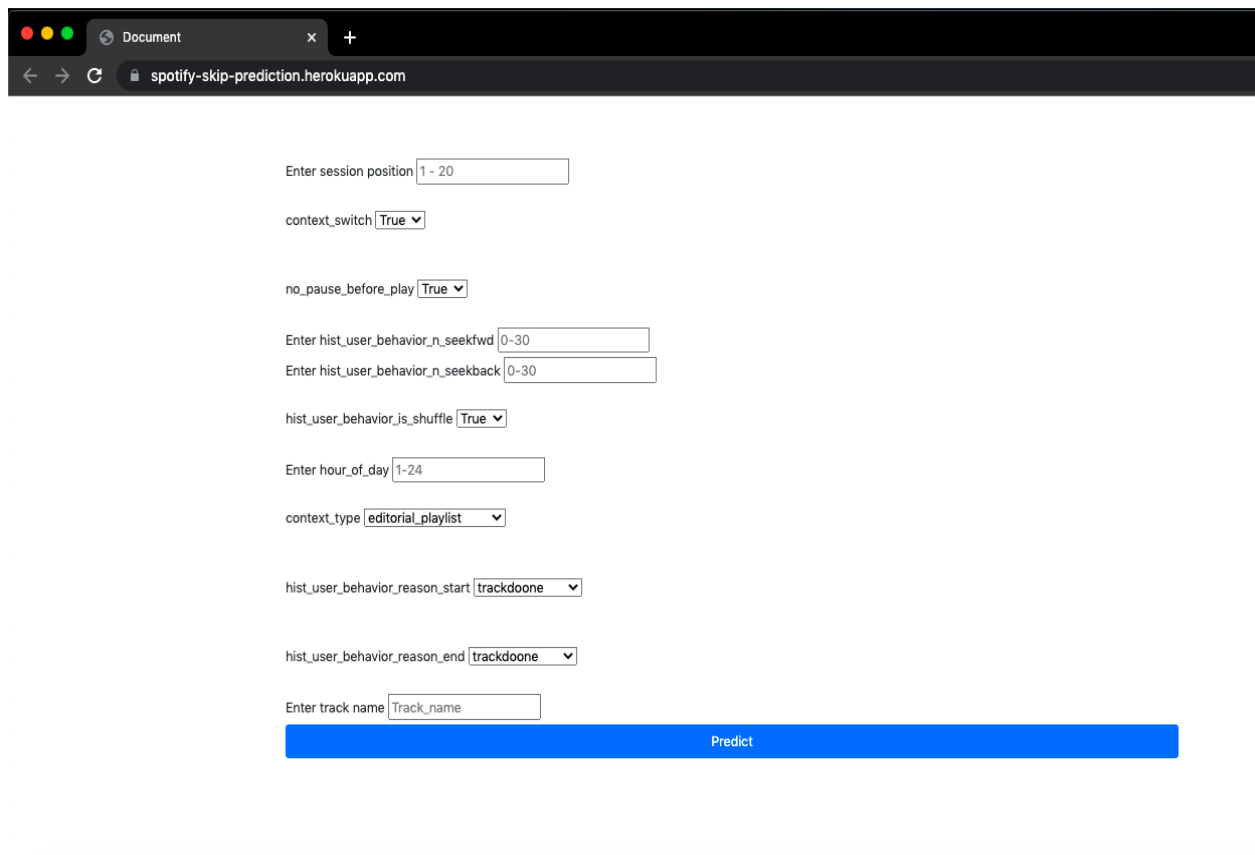
Support vector machines are sets of supervised learning methods used for classification and regression problems. The main reason for using SVM in our project is it works very well with high dimensional spaces for SVM we got an accuracy of 87%

We use many default parameters and linear in the kernel for training the model

## Deployment:

Here I use called Flask framework and Heroku cloud for the deployment. And I created a simple frontend using HTML and bootstrap.

The main challenge for deployment is the webserver I have to get the acoustics features of the song based on the name of the song for this first I got the track-id of the song in response by sending the name of the song as a request and further I send a request for acoustics features based in track ids from Spotify WEB API



The screenshot shows a web browser window with the address bar displaying "spotify-skip-prediction.herokuapp.com". The page contains a form with the following fields and controls:

- Enter session position:
- context\_switch: ☒
- no\_pause\_before\_play: ☒
- Enter hist\_user\_behavior\_n\_seekfwd:
- Enter hist\_user\_behavior\_n\_seekback:
- hist\_user\_behavior\_is\_shuffle: ☒
- Enter hour\_of\_day:
- context\_type:
- hist\_user\_behavior\_reason\_start:
- hist\_user\_behavior\_reason\_end:
- Enter track name:
- A blue button labeled "Predict" is located at the bottom of the form.

## **Conclusion:**

Building a sequential skip prediction was quite challenging. This project is very beneficial for streaming services like Spotify on their recommended system. The following accuracy of prediction is obtained using Machine Learning methods like LGBT and SVM, which is 87.5% and 87.07% respectively.

We can further improve this project using seq2seq Lstm and attention models to get the best accuracy and provide the best user experience on the platform.