,

Spotify Popularity Predictor

Pablo Assoni Raiter

Abstract

An independent record label is aiming to improve their artists album sales. To do it, they intend to apply Machine Learning in order to sculpt their songs in a more comercial way. The first attempt to do it is using Spotify Popularity feature. This feature measures a song success based on how many listenings a song had. The label firmly believe that some features can be good predictors of a song success and they want to know what are them. Also the label wants to predict with maximum accuracy how well a song will perform on Spotify every time a new song is releasen. We will create a model based on 19k songs available on Spotify and its main features, such as energy, key, loudness and many others.

1 Domain background

Music business is getting more and more competitive. New record labels are being created day after day and everyone wants to be on the top charts. More recently Machine Learning became part of the music market. Record labels are using algorithms to find the features that make a song popular. Also record labels are using these algorithms to predict if a song will be a hit or not, i.e. its popularity. Mostly music streaming platforms are providing data about songs features and its popularity. Pham, Kyauk and Park (2015) recently published a study providing results about relevant music popularity attributes. In this sense, we want to validate its results and create a model capable of predicting a song popularity based on Spotify data.

2 Problem statement

Our independent record label wants to achieve more success with its artists songs. In this project we aim to create a model capable of detecting key features for a popular song and capable of predicting if a song will be popular or not. To do it, we will use as a target Spotify popularity feature. This feature is an index based on how many plays a song had and how recent those plays. To create our model, we also intend to use Spotify data. We intend to work with the following features: key, valence, tempo, speechiness, energy, loudness, and many others. Our dataset need to be treated to serve as input to our model. As a benchmark,

we will use Linear Regression, one of the simplest models available on Machine Learning. After that, we will try to test several models in order to find if we can beat Linear Regression or not. We expect to reduce our error to the maximum and to provide a model that really fits the data and give insights about it.

3 Datasets and inputs

In this project we will use a dataset available on Kaggle called "19000 Spotify Songs". You will find the dataset in: https://www.kaggle.com/edalrami/19000-spotify-songs. This dataset contains It contains several features available on the platform that may be interesting to predict a song performance. Initially we intend to use 13 features: danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, duration and time signature. As expected, our dataset contains over 19000 songs observations, containing a few missing values. We believe it will be enough to fit a regression model capable of providing consistent results. Our data is not previously treated and need some processing. Also we will need to divide our dataset into train and test datasets in order to test our model without biased data.

4 Benchmark model

We have chosen Linear Regression as our benchmark model. Linear Regression is the most popular and known Machine Learning model and it is a good benchmark model to verify if more complex models really provide more accurate results. Because of its simplicity it is probable that our model will not perform very well on our complex dataset. Since we are not really trying to look for causality in our variables and we are aiming just to provide a main benchmark predictor, we believe that many statistical hypothesis could be dropped with the purposes of predicting only. Anyway we intend to at least provide normalized data to our model, since it is very well-known that General Linear Models handle data better if it is normalized. A Linear Regression formula is provided below:

$$y = x_1\beta_1 + x_2\beta_2 + \ldots + x_i\beta_i + \varepsilon \tag{1}$$

5 Solution statement

In order to beat our benchmark model, we will test the following Machine Learning models: (i) K-Neighbors Regression, (ii) Lasso Regression, (iii) Ridge Regression, (iv) Decision Tree Regression, (v) Random Forest Regression, (vi) Extra Trees Regression, (vii) Gradient Boosting Regression and (viii) Adaboost Regression. The best way to search for a perfect Machine Learning model for the data we are working is simply to test for a bunch of them.

Since we are working on AWS SageMaker environment, it will be easier than using local computational power.

6 Evaluation metrics

In order to evaluate our model, one metric will not be enough to see its performance. We will use four known metrics to get the properly results: Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE) and R-Squared (R2). All these metrics serve for different purposes and address model performance in different ways. In this way we can definitely validate if our model will beat the benchmark Linear Regression model: we can use MAE, MSE and RMSE to verify the model mean error (its mean variance from the real result) and we can use R-Squared to verify the model fit. If our MAE, MSE and RMSE presents low values we are closer to a consistent model. If our R-Squared is closer to one, it means that our model is having a good fit to the data (if we do not put irrelevant data to the model, of course). The metrics formulas are provided below:

$$MAE = (\frac{1}{n}) \sum_{i=1}^{n} |y_i - x_i| \tag{2}$$

$$MSE = (\frac{1}{n}) \sum_{i=1}^{n} (y_i - x_i)^2 \tag{3}$$

$$RMSE = \sqrt{(\frac{1}{n}) \sum_{i=1}^{n} (y_i - x_i)^2} \tag{4}$$

$$R2 = SSR/SST \tag{5}$$

In R-Squared formula SSR means "Sum of Squares of Regression" and SST "Total Sum of Squares".

7 Initial project design

Initially we have pre-processed data (transformed datatypes, dropped irrelevant columns and missing values observations), dealt with categorical columns and normalized it using MinMaxScaler. Initially, we did not find relevant columns to be dropped. Even if they are well correlated, they do not transmit the same information. Our data is ready to be sent to Machine Learning algorithms listed above in Section 5.

References

[1] James Pham, Edric Kyauk, and Edwin Park. *Predicting Song Popularity*. Stanford University, 2015.