Music Popularity Prediction Supervised Machine Learning

Advanced Data Analysis
University of Lausanne

Neil Touafek
17818832 neil.touafek@unil.ch

Khaled Chebbi
16816712 khaled.chebbi@unil.ch

I. Abstract

Music plays a major role throughout the evolution of humanity, from a motivational aspect to a therapeutic one, it is in the heart of our societies and this as far as the human can remember. Music has the power of uniting people around it and makes the audiences agree on the level of goodness of the music and this gives value and a recognition to it. But what makes a music universally recognized ? There must be features and attributes that are in common between more or less popular songs. Nowadays and with the continuous evolution of multimedia applications a lot of research effort is done to increase the quality of the services provided by platforms such as Spotify. The increase in the amount of Music Data available via this platforms enhance the Machine Learning research around it and give the opportunity to provide a more or less accurate prediction of the popularity of a song using Machine Learning tools. Therefore, this paper aims to create and compare some algorithms that provide a prediction on yes or no a song tend to be popular based on aggregated features.

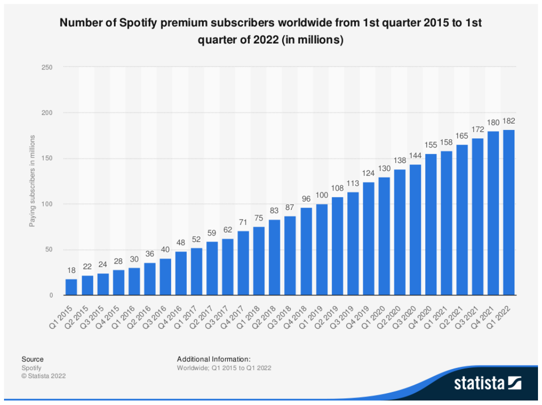


Fig. 1. Spotify Users evolution from 2015 to 2022, Statista

II. Introduction

During the last decade, the explosion of the multimedia platforms such as Spotify or Netlflix as we can see in Figure 1 offered a new path and opened new horizons for the understanding of this big amount of Data in term of size and diversity. With the gigantic amount of revenues that this multimedia companies are making it became more and more important for them to understand the behavior of their audiences towards their content. Therefore Multimedia Retrieval Systems (MRS) became an important subject for both research and effective applications that improve the overall experience and more precisely the large-scale data management process.

Multimedia Retrieval Systems (MRS) is the research discipline of computer science that aims at extracting semantic information from multimedia data sources. It includes audio images videos and text sources. In the recent years this field of research became more and more important showing accurate results either in a multi-modality way combining more than one approach or using a single one. Research showed that multimodality approaches are more efficient in Information Retrieval processes.

In this field and more specifically the Music Information Retrieval exploded with the raise of platforms such as spotify or Lastfm. Computer science and MIR plays tho a crucial role supporting theses platforms in several objectives. Either music genres classifications for playlists creations with similar songs or song suggestions and recommendations, popularity predictions and even music generation, powerful algorithms are needed in the field and asked from the multinationals to provide a better customer experience and a greater revenue for them.

Throughout all theses applications, music popularity prediction comes out as one of the most important applications. When the global music market was worth 26bn in 2021 and the majority of the revenues is generated by popular songs , having a deep understanding of what makes a song popular has big implications on the parties involved in the music industry.

Setting what makes a song popular can help the producers to direct their productions in the right way and it also can be used to understand the taste and to profile their audiences and improve their recommendations for customized music suggestions.

Moreover with the rise of Machine and Deep Learning tools the last 10 years, referred as the Hit song science when it comes to music applications, many companies and researchers works closely to solve complex classification and pattern recognition problems. Having the ability to go over most of the non-linear complex problems by adapting its parameters according to the data available.

Available tools combined provide an important way to improve the popularity music prediction and MIR problems more generaly.

In this paper we aim to provide a comparison between both supervised machine and deep learning using from simplest to more sophisticated classification models. Most of the research when it comes to music popularity prediction uses a single modality approach for example audio features or lyrics features. Our dataset allow us to cross several categories and predict popularity based on a multi-modality dataset. We are provided features for more than 100K tracks mainly from spotify: Such as liveness and danceability of the track. Extracted audio features: Such as Tonnetz and spectral compositions. Artists informations: Such as artist popularity and number of followers. And finally lyrics features: Such as the number of words in the song. Our data varies from numerical to binary and objects types.

We try to solve and predict song popularity and its corresponding error based on these features using Machine learning classification algorithms like Random Forest and Decision Tree classifier, and deep learning classification networks like multi layers perceptron and neural networks.

The following paper is organized as follows: In section 3, a description of the research question and the relevant literature, in section 4 the methodology applied to address the research question, in section 5 a description of the data set, in section 6 discussion of the implementation, in section 7 the results and finaly the conclusion. .

III. Description of the Research Question and Relevant Literature

The Hit Song Science is the science that concerns the possibility of predicting the popularity of a song before it's distribution using machine learning. In their study [1] has shown that audio features can indeed be used to outperform a random oracle when predicting top 10 versus top 30-40 hits.

The last years HSS was at the core of several studies that aims to provide an accurate prediction of the song popularity using multi and single modality features. This research are an important benefit for all the industry. From the consumers to the producers and the multimedia platforms.

Understanding the features of a song that makes it a hit is important as it provide a pattern and characteristic to base on music production that will give a higher probability of hitting and to attract audiences.

Audio and lyrics features are in the core of the studies that concern the popularity predictions. Some studies gather even more factors that concern for example cultural aspect and psychological ones as it is done in the following study [2]

Other interesting features to consider are social network and concert and festival data for example. In their study [3] used theses features in addition to the songs acoustic features. This work aimed at predicting the popularity of the artist and applied a logistic regression with a maximum accuracy reached of 39 percent.

In their study [4] and similar to our classification problem the authors used several classification machine learning models such as logistic regression a support vector machine and applied their models to The Million Track Dataset [7] which provide several features about tracks.

In addition to that several works uses deep learning tools to analyze signals of the song by studying it as an image by using convolutional neural networks as in Using Deep Learning to Predict Hip-Hop Popularity on Spotify ending up with an accuracy of 0.58.

In our work we will perform several models selected based on this last research listed before and using The SpotGenTrack dataset that gather information from powerful platforms that are spotify and Genius and adding other audio features. The aim of the creation of this dataset is to fill the lack of information that some studies were not taking into account and performing a multi modality solution to the problem.

IV. Methodology and Prediction Models

In this section we are going to show the prediction models that we used to increase and maximaze the accuracy of our prediction and to compare different ways to reach our goal.

A. *Decision Tree*

Decision tree classifier is a primitive machine learning model that is non-parametric supervised learning method that goes from observations about an item to conclusions about the target. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

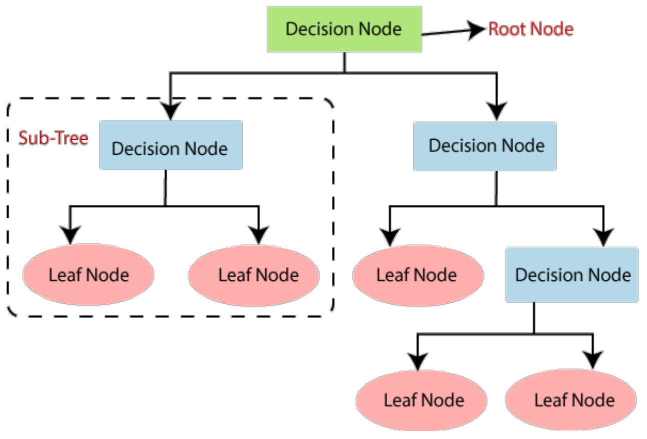


Fig. 2. decision-tree-classification-algorithm

B. *Random Forests*

Random forests is an ensemble learning method for classification, regression and other tasks that operates by fitting a multitude of decision trees classifiers. For classification tasks, the output of the random forest is the class selected by most trees. Random decision forests correct for decision trees habit of overfitting to their training set by using averaging.

C. Logistic Regression

The logistic regression is a classifier that uses a logistic function more specifically named sigmoid function that transform numbers between 0 and 1. The model uses an equation as the representation where the input values are combined linearly using weights to predict the output y . The output value is modeled as a binary value that determine the class of the output 0 or 1.

D. Adaptative Boosting

An AdaBoost classifier is an meta-estimator similar to the concept of random forest that begins by fitting a tree classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where there is a correction on the misclassified objects by adjusting the individual weights assigned to the training samples. The model is based on a repeatedly modified versions of the data and it is so called the boosting iterations. The prediction from all of the iterations are combined through a weighted majority cote to produce the final prediction.

E. Gradient Boosting

GB builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage n classes regression trees are fit on the negative gradient of the loss function, e.g. binary or multiclass log loss. In our case and for binary classification only a single regression tree is induced.

F. mlp

A multilayer perceptron is a connected class of artificial neural network (ANN). An MLP is composed of layers of nodes: an input layer, a hidden layer and an output layer as a minimum. Each node uses a non linear activation function. MLP utilizes a supervised learning technique called backpropagation for training. It can distinguish data that is not linearly separable.

G. DNN

Build on a similar way as the Logistic Regression the the deep neural network is a collection of connected layers with multiple activation functions. In our model and with ou large-scale features dataset we constructed a neural net of 7 hidden layers of decreasing number of neurons, from 512 to 8 neurons. As an activation function we use Rectified Linear Unit(ReLU) to activate the 7 first layers. The last layer uses a sigmoid function that produce a value between 0 and 1 and that perform our classification problem. As our model quickly overfitted for the first trials we added some configurations to reduce overfitting. We added a dropout regularization to our deep neural network with a value of 0.2 which means that that we drop randomly a ratio of 0.2 of the nodes during the training at each stage.

H. cnn

Convolutional neural networks are build in a similar way that the neural networks in the sense that they are made up of hidden layers consisting of neurons with “learnable” parameters. These neurons receive inputs, perform a dot product, and then follows it with a non-linearity. This type neural network is essentially used for image classification. We are aware that this type of neural network is not appropriate for this type of data but we wanted to give it a try.

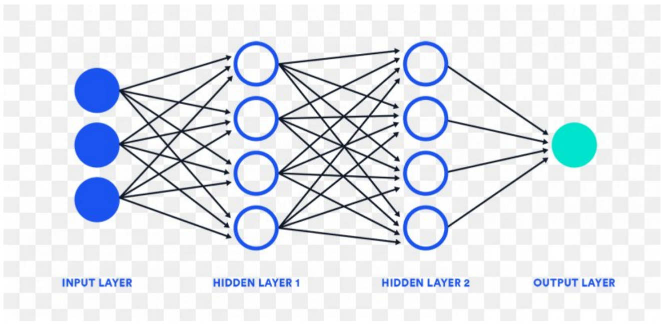


Fig. 3. Architecture of a neural network

I. hypertuning

In a matter of optimization we are trying different parameters to our models which is useful in adapting this models to our dataset.

We introduce to our models a bunch of hyper parameters such has the optimizer or the length of our decision tree classifier.

There is many ways to select the hyper parameters that performed the best with our models but in our case we are using a grid search model selection. It’s an exhaustive search over the specified parameter values that we provide. The set of parameters are optimized by cross-validation over the parameter grid.

V. Description of the Data Set

Initially we took a look on several datasets constructed mainly with a single modality approach that aim to provide research material for prediction. Most of these works get their data from Spotify API which provide a very complete informations regarding acoustic features and the artists.

Our Dataset SpotGenTrack is a collection of multi-modal features constructed by that gathered acoustic features from Spotify and audio and text features extracted straight from songs . This dataset is presented as an alternative solution to existing datasets that will facilitate researchers when comparing and promoting their models. In our paper we are going to give information about how this features were collected.

For convenience matter and accuracy target we deleted some features that we didn’t use in our models because we noticed that working with high dimensionality was not adequate and gave us better results after running our models.

The dataset is a concatenation of 4 csv files constructed in different ways and the description of all our used variables in each modality are listed below.

A. High level audio features

Spotify provides a good starting point for popularity prediction. They give access with their API to several acoustic features that are wildly used by developer in music genre classification and popularity prediction. The table below shows the details about each features.

Audio Features	Description
Acousticness	A confidence measure $\in [0,1]$ of whether the track is acoustic.
Danceability	A value $\in [0,1]$ that describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity..
Duration	The duration of the track in ms
Energy	A confidence measure $\in [0,1]$ that represents a perceptual measure of intensity and activity.
Instrumentalness	A measure $\in [0,1]$ that predicts whether the track contains no vocals.
Key	The key the track is in. Integers map to pitches using standard Pitch Class notation $\in \{0,1,2,\ldots 11\}$.
Liveness	A value $\in [0,1]$ of whether there is presence or not of an audience in the recording and indicates a likelihood.
Loudness	The overall loudness of a track in decibels (dB).
Mode	Indicates the modality (major or minor) of a track. Major is represented by a 1 and minor by a 0.
Speechiness	A measure $\in [0,1]$ of whether the presence of spoken words in the track is or not detected.
Tempo	The overall estimated tempo of a track in beats per minute (BPM).
Valance	A measure $\in [0,1]$ describing the musical positiveness conveyed by a track where high valence indicate sound positive and lower valence indicates sound more negative.
Popularity	A measure $\in [0,100]$ describing how popular a track is, where 0 means no popular and 100 very popular.

Fig. 4. Table features from spotify API

B. Lyrics features

The lyrics features extraction was done upstream by the authors of this dataset to set the descriptors and recognize patterns in the lyrics. NLP techniques and a stylometric analyses were performed to obtain the following features:

1) **The total number of sentences**
2) **the average number of words per sentence**
3) **the total number of words**
4) **the average number of syllables per word**
5) **sentence similarity coefficient**
6) **a vocabulary wealth coefficient**

In the following parts we are going to explain how the sentence similarity coefficient and vocabulary wealth coefficient were obtained.

1) **sentence similarity coefficient** To understand the sentence similarity coefficient we need to set some concepts.
   In information retrieval, tf–idf, short for term frequency–inverse document frequency, is a statistical value that aims to reflect how important a word is to a document in a collection in our case the words in the sentences of the lyric. The tf–idf value increases proportionally to the number of times a word appears in the sentence and is offset by the number of sentences in the lyric, which helps to adjust for the fact that some words appear more frequently in general. In the algorithm 1 we can see how they obtained the sentence similarity coefficient. This aim to recognize repetitive patterns in the lyric that might be correlated with the song popularity. The coefficient is bounded from 0 to 1. A value of 1 means that all the sentences in the lyric are the same. The following algorithm shows how the calculation of the coeficient was done.

Algorithm 1 Sentence Similarity Coefficient

```
1:  procedure SentenceSimilarity(lyric)
2:      if |lyric| > 1 then
3:          $l \leftarrow lyric$
4:          $tf\_idf \leftarrow computeTfIdf(l)$
5:          $m \leftarrow computeCosineSimilarity(tf\_idf)$
6:          $diag \leftarrow getUpperDiagonal(m)$
7:          $t \leftarrow |diag|$
8:          $n \leftarrow 0$
9:          for s in diag do
10:             if s $\geq \mu$ then
11:                 $n \leftarrow n + 1$
12:         $sim \leftarrow n/t$
13:     return $sim$
```

Fig. 5. Sentence similarity algorithm

2) **Vocabulary wealth coefficient** The vocabulary wealth coefficient is calculated by defining a set of words present in a lyric. From this set a subsequent set is created from the distinct words present in the corpus and this determine the vocabulary length of the corpus. A set of non-repetitive words is created and this lead us to the ratio of non repetitive words over the total number of words in the lyric. The coefficient is bounded from 0 to 1. A value of 1 means that the song is diversified which means that all the words are different.

C. Audio features

In this part we are going to explain some of the extracted feature from the song signal.

1) **Zero crossing rate** The zero-crossing rate (ZCR) is the rate at which a signal changes from positive to zero to negative or from negative to zero to positive. Its value has been widely used in both speech recognition and music information retrieval, being a key feature to classify percussive sounds. Additionally, it is well-known that periodic sounds yield to a smaller ZCR whereas noisy sounds tend to have a larger value.

$$zcr = \frac{1}{T-1} \sum_{t=1}^{T-1} 1_{\mathbb{R}_{<0}} \left(s_t s_{t-1} \right) \tag{1}$$

where s is a signal of length T and $1_{\mathbb{R}_{<0}}$ is an indicator function.

2) **Spectral centroid** The spectral centroid is a measure used in digital signal processing to characterise a spectrum. It indicates where the center of mass of the spectrum is located. Perceptually, it has a robust connection with the impression of brightness of a sound. It is calculated as the weighted mean of the frequencies present in the signal, determined using a Fourier transform, with their magnitudes as the weights:

$$\text{Centroid } = \frac{\sum_{n=0}^{N-1} f(n) x(n)}{\sum_{n=0}^{N-1} x(n)} \tag{2}$$

where x(n) represents the weighted frequency value, or magnitude, of bin number n, and f(n) represents the center frequency of that bin.

3) **Spectral bandwidth** The bandwidth of a signal is defined as the difference between the upper and lower frequencies of a signal. It is measured in terms of Hertz(Hz) the unit of frequency.

$$SB = \left(\sum_{k=1}^{K} S(k)(f(k) - \zeta)^p \right)^{\frac{1}{p}} \tag{3}$$

where S(k) refers to the spectral magnitude at frequency bin k, f(k) indicates the frequency at bin k and refers to the spectral centroid.

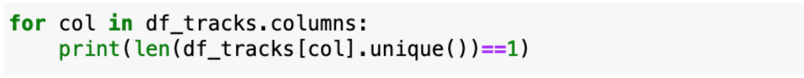
VI. Discussion of the implementation

Our code starts with the load of the csv files using pandas library. The datasets are loaded in 4 different data frames which are dftracks, dfartists, dflyricsfeatures and dflowlevelaudiofeatures.

Our data frame are from different types. We have sequences strings objects and int values. We started by dropping some of the columns that were not needed or at least we presumed they will not affect our output.

We started by taking off this list of columns name,tracknumber,discnumber,type,Unnamed 0 , analysisurl , href , previewurl , trackhref , uri, tracknameprev,albumid,lyrics,country.

As we have all the lyrics features needed for our study we didn't keep the lyrics column.



```python
for col in df_tracks.columns:
    print(len(df_tracks[col].unique())==1)
```

Fig. 6. Columns1

With this algorithm we checked if there is some columns that all the values for the rows are the same which undoubtedly will not have consequences on our output.

The name of the track is not needed as we have the id of the track to merge with the other dataframes. We added a column in our data frame that count the number of markets that the track is available in. For the country of the track we found that there is only 2 countries which doesn't make sense for our prediction, so we took off this column. In a nutshell we kept the id track that we set as an index and the acoustic features provided from spotify, plus other information about the track like the duration the available markets and the popularity our target.

The popularity rate is a value between 0 and 100, 100 meaning that a song is very popular.

From the lyrics data frame we kept all the columns as they were provided by SpotGenTrack. We only took off an index column that was not needed.

The artist dataframe provide informations about the artist . His popularity the number of followers that he has, his id ,the genre, his name and other information. From this dataset we kept only the id, the popularity, the number of followers and his music genres.

From the low level audio features dataset we kept in the beginning all the columns of the data frame. After performing our models at the end we decided to take off some of the columns that were decreasing our performance. After taking off this columns we increased our accuracy by 3 percent in average in all our models. We presume that this is not the right way to do it we could have encode and reduce our data frame by keeping all this information. Neverthleess when we computed the correlation between these features and ou popularity they were not signficant.

Our first merge of data frame if done between the tracks and the lyrics features as they have the same indexes that is the track id. For this we performed a inner join on the dftracks index. We merged after that this last data frame with the low level features with a also a inner join which allow us also to delete the not assigned rows.

So far we have merged 3 of our csv files with the id track as an index. In the next part we aimed to merge our last merged data frame with the artist information on the artist id. Our merged data frame contain the artist id but the sequences contain characters that are not in the artist dataframe. To perform this we replaced this characters by using the replace function and so the artist id sequence matches with the artist id in the artist dataframe.

We noticed also that there is some songs with different artists and we were not able to merge it with our artist dataframe so we putted off all the tracks with more than 1 artist. Still as our set of observations is enough big to perform the models. Here the problem was that we had to assign for each track an artist popularity the number of followers of the artist and the genre of the artist. as we had songs with different artist we were thinking of assigning a weight to the popularity or to assign the higher artist popularity to the song. We decided that as we are not loosing a lot of data to drop the rows with more than 1 artist.

We have now gathered all our features in one dataframe.

Only one problem remains which genre are we going to assign to each track as when we did the last merge the artists might have several genres and tho the tracks.

The problem is that we have in our dataset pop songs from canada and pop song form uk and so the genre assigned are respectively canadian pop and uk pop. We supposed that this genres has to be in the same genre which is pop. To solve this problem we started by writing an algorithm that calculate the most appearing genres in our dataset. First we had more than 1000 different genres so we had to concatenate them. Ou algorithm selected us the 20 most common genres in our dataset and we wrote a function that assign genres to each track. From a list also customized with help of wikipedia we extracted the main music genres.

Here is the list of our selected general genres followed by the algorithm to assign them pop,rap, edm,rock, hiphop, house, rb, latin, jazz, classic, folk, blues, country, reggae, soundtrack, metal, other genres as you can see in Fig.7.

Basically if from the list of genre that we assigned to each track by merging with the artist genres, there is one of the main genres we assign this main genre to the track. All the other genres are assigned as other genres. So the canadian pop and uk pop will both appear as a pop song.

After that we performed an algorithm that encode our genres so they appear as binary columns in our dataset and we applied our model to all the genres. So for a track that is a pop track it will show a 1 in pop column.

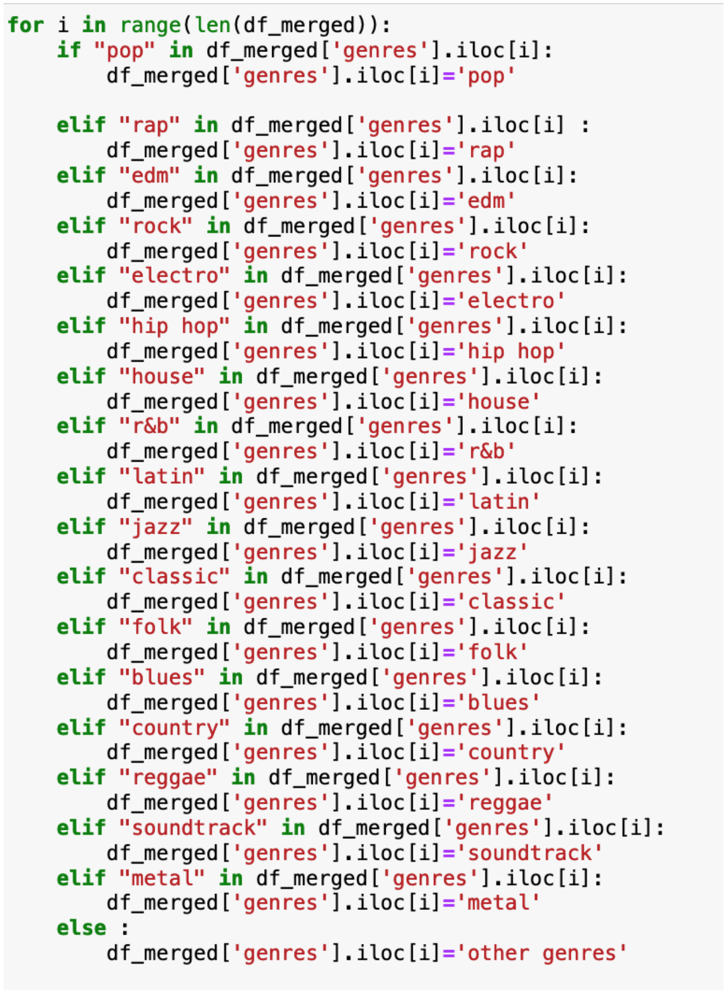
In our main code we also checked for the outliers.

We added an algorithm that that calculate the Z-score of each column and we added zscore columns for each feature in a copy of our dataframe. The zscore formula is showed below:

$$\text{Cote}\, Z = \frac{X-\mu}{\sigma} \tag{4}$$

We based our assignment of the outliers as it's done in most of the similar research projects. We will drop the row if the absolute value of the zscore is higher than 3. Our algorithm return the index of the rows that are considered as outliers. And then we use a function to drop this rows as shown in the following algorithm.

To get description of our dataset we used a pair plot for all our features and we also plotted each distribution alone to understand how our data distributed. Most of our values are bounded form 0 to 1 so we don't need to further standardization for this features and they are ready to be provided to our models.



```python
for i in range(len(df_merged)):
    if "pop" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='pop'

    elif "rap" in df_merged['genres'].iloc[i] :
        df_merged['genres'].iloc[i]='rap'
    elif "edm" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='edm'
    elif "rock" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='rock'
    elif "electro" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='electro'
    elif "hip hop" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='hip hop'
    elif "house" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='house'
    elif "r&b" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='r&b'
    elif "latin" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='latin'
    elif "jazz" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='jazz'
    elif "classic" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='classic'
    elif "folk" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='folk'
    elif "blues" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='blues'
    elif "country" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='country'
    elif "reggae" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='reggae'
    elif "soundtrack" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='soundtrack'
    elif "metal" in df_merged['genres'].iloc[i]:
        df_merged['genres'].iloc[i]='metal'
    else :
        df_merged['genres'].iloc[i]='other genres'
```

Fig. 7. Genres assignation algorithm

To visualise our data we also plotted regplots with seaborn to show eventual dependencies between our dependent variable and our features. We also used a heattmap (correlation map) with seaborn to check for eventual correlation. From this we can for example notice that there is a high correlation between the popularity of the song and the popularity of the artist.

To scale our data we performed a check for normality by plotting all the distributions of all our features. We noticed that some features are not scaled so we performed a minimal scaling algorithm to range our values between 0 and 1.

For the data that are presented as classes for example the key or the time signature we one hot encoded them using our encoding algorithm.

Our final dataframe contains at the end only numerical values ranged from 0 to 1 for the continuous values and binary columns for the discrete ones. We ended up with 27 continuous columns and 36 binary columns. We decided to use a cut off function that labels our popularity. We decided to assign to labels 1 and 0 for the popular and not popular songs. We used the cut function to perform this task. It give a value of one for the tracks that have a popularity higher than the mean popularity and 0 otherwise. The cutoff is around 40 in popularity value. Basically if a song has a popularity higher than 40 it's considered as a popular song.

Model implementation. To implement our models we decided to approach the model in 2 different ways. First, for each of the models we performed a basic training and fitting without any specified parameters. This constitute our baseline to compare with further applications. The output of our models except for the neural networks models are arrays of 0 and 1 values.

To select the best parameters and to perform a cross validation we used a grid search except for our dnn,mlp and cnn models .

We used a class report function for each of the models including the neural networks that gives us the accuracy the precision the recall the F1 score the mean squared error and the mean absolute error. We also generate a confusion matrix for each of the models.

Our Dnn model is composed of 1 first layer as input that is constituted of 512 nodes with a ReLU activation function. Follows 6 hidden layers decreasing with a ration of 0.5 also using a ReLU activation function.

Our last layer is 1 node layer with a sigmoid activation function that provide us the probability of each track to be popular or not popular. Basically our output is a probability that is higher than 0.5 the track is popular and not popular otherwise. We use a learning rate of 0.0001 with empirical trials and a callback function of early stopping with loss value as monitor and a patience of 5. We also use an Adam optimizer which is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments. We then plot our accuracy and validation accuracy and loss and validation loss for both convolutional and deep neural networks.

We also used a convolutional neural network that noramlly is not used for this type of classification. In other studies we use this Neural network for image classification. We had to transform the input size by using a reshape and introduced a kernel of shape 1;63. We used a max pooling size of 1;6 and same padding with a ReLU activation function. After verification we could have changed our 2D convolutional NN to 1D convolutional NN that is more appropriate.

1) **ReLU**: $y = \begin{cases} x & \text{if } \ x > 0 \\ 0 & \text{otherwise} \end{cases}$

with gradient: $\frac{\partial \mathcal{L}}{\partial x} = \begin{cases} \frac{\partial \mathcal{L}}{\partial y} & \text{if } \ x > 0 \\ 0 & \text{otherwise} \end{cases}$

2) **Sigmoid**: $y = \sigma(x) = \frac{1}{1+e^{-x}}$

with gradient: $\frac{\partial \mathcal{L}}{\partial x} = \frac{\partial \mathcal{L}}{\partial y}\sigma(x)(1-\sigma(x))$

VII. Results

After applying our models to our cleaned dataset our results are pretty interesting. As a starting point we would infer that our models performed a good accuracy on our dataset. The best accuracy that we reached is with our random forests models with a score of 0.7737 and an error of 0.22. In this part we are going to show all our results for each of the models and with the different parameters that we have introduced .

A. *Decision Tree*

Approach	Result
Baseline	0.70623
Searchgrid	0.75305

B. *Random Forests*

Approach	Result
Baseline	0.7737
Searchgrid	0.7648

C. *Logistic Regression*

Approach	Result
Baseline	0.7501
Searchgrid	0.7509

D. *Adaptative Boosting*

Approach	Result
Baseline	0.7566
Searchgrid	0.7566

E. *Gradient Boosting*

Approach	Result
Baseline	0.7684
Searchgrid	0.7695

F. *mlp*

Approach	Result
Baseline	0.7551
Searchgrid	0.7690

G. *DNN*

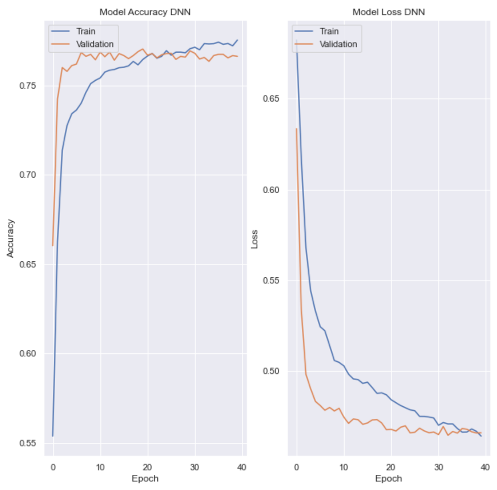


Fig. 8. dnn accuracy and loss with validation data

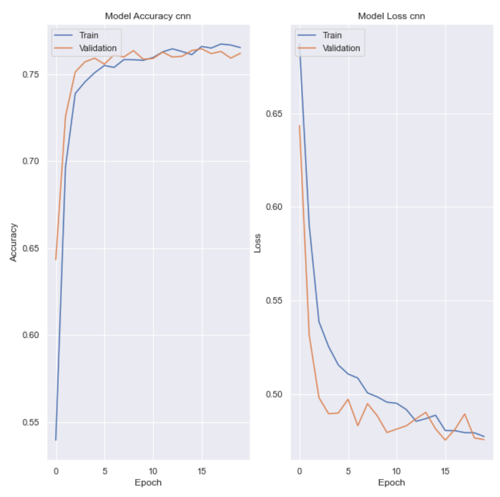


Fig. 9. Cnn accuracy and loss with validation data

H. cnn

I. Table of results

Models	Accuracy	precision	recall	F1	score	mse
Decision tree	0.7530	0.7002	0.8452	0.7659	0.2469	0.2469
Random forests	0.7737	0.7388	0.8146	0.7748	0.2262	0.2351
Logistic Regression	0.7509	0.7268	0.76742	0.7466	0.2262	0.2490
AdaBoost	0.7566	0.7139	0.8193	0.7630	0.2433	0.2433
GradientBoost	0.7695	0.7317	0.8177	0.8177	0.2304	0.2304
MLP	0.7595	0.7243	0.8023	0.7613	0.24046	0.2404
DNN	0.7538	0.7315	0.7663	0.7485	0.2461	0.2461
cnn	0.7541	0.7088	0.8244	0.7622	0.2458	0.2458

VIII. Conclusion

A complete description of the models and a prediction of the song popularity was performed throughout this paper. As our dataset provided us enough informations and diversity in the features that led us to an accuracy score enough significant. However we noticed that our accuracy are much highr than the accuracies obtained in other studies. Which lead us to ask questions about our proceeding our process of cleaning the data and our models implemented. However we believe that this dataset is well constructed and gather way much more features than others. Which also let us think that this is the reason of our relatively high accuracy. Our objectif of predicting the popularity of the track is reached and in futures project we aim to perform our convolutional neural network on the signal song straight. We aim also to perform a better deep neural network and to see more closely to our random forest model as it performed the best accuracy score. We also aim to try our models on a test songs that we want to extract the signal features from it and develop the algorithm for.

The code to test the model is available in the folder seent with this paper.

References

[1] Herremans, D., Martens, D., Sörensen, K. (2014)."Dance hit song prediction".(PDF).Journal of New Music Research.43 (3): 291–302.

[2] Ruth Dhanaraj and Beth Logan in Automatic prediction of hit songs in 2005.

[3] Arakelyan et al. ARAKELYAN, S. et al. Mining and forecasting career trajectories of music artists. CoRR, abs/1805.03324, 2018.

[4] James Pham, Edric Kyauk, and Edwin Park. Predicting song popularity. nd): n. pag. Web, 26, 2016.

[5] Nicholas Indorf https://towardsdatascience.com/using-deep-learning-to-predict-hip-hop-popularity-on-spotify-1125dc734ac2

[6] DAVID MARTÍN-GUTIÉRREZ , GUSTAVO HERNÁNDEZ PEÑALOZA , ALBERTO BELMONTE-HERNÁNDEZ and FEDERICO ÁLVAREZ from Universidad Politecnica de Madrid

[7] Million track dataset