Hit Song Prediction

Team 11

Parth Chhabra
2019069

Samyak Jain
2019098

Sarthak Johari
2019099

Yash Mathne
2019129

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY **DELHI**

Introduction

Introduction

- With the forever-expanding music industry and the number of people keen on listening to popular music, it becomes very important to come up with a classifier that can predict whether a song is ‘hit’ or ‘non-hit’ to help the musicians and the music labels.
- Using the data collected from Billboard, Spotify and Million Song Dataset, our project takes into account several features for a song like audio features and related artist data and based on that uses Machine Learning based classification algorithms to develop models that can help us achieve the desired classification.
- Our goal is to make accurate as well precise predictions.
- Our models will indicate what choices of a particular feature make a positive impact so that the musicians and music engineers can plan accordingly to give their songs the best chance of being classified as ‘hit’.

Literature Review

Literature Review

Pachet, Francois & Roy, Pierre. (2008). Hit Song Science Is Not Yet a Science..

- It used external (extracted from the musical ecosystem such as social media presence) and internal features (extracted from the audio) to predict a song's popularity.
- Also made use of 632 manually labelled features for each song to encapsulate all the internal and external features.
- Could not develop an accurate model and concluded it could not be done by the state of the art machine learning.

Literature Review

Ni, Yizhao et al. "Hit Song Science Once Again a Science?" (2011)

- It focused on using low-level internal features to predict a song's popularity.
- It used a classifier which was a function of time along with a shifting perceptron learning agent. Significantly outperformed a random oracle and obtained a 60% accuracy with it's predictions.
- However, it was limited to UK's billboards and may not generalise well.

Zangerle, Eva et al. "Hit Song Prediction: Leveraging Low- and High-Level Audio Features." (2019)

- The paper focused on using low-level and high-level internal features to predict the commercial success of a song.
- It added the date as a high-level feature to contextualise the song temporally to improve accuracy.
- Also, it used wide and deep network regression to obtain a 75% accuracy.

Dataset: Details, Preprocessing & Analysis

Dataset Extraction

- We have used a one-tenth subset of one million songs extracted from Million Song Database.
- We use spotipy to query Spotify API and get information of 29,371 songs released between 2006 to 2020.
- From Billboard charts and Spotify API we get audio feature information for 3,796 billboard hit songs release between 2006 to 2020.
- We also explored a different form of data i.e - audio data to train CNN models and perform the classification task.
- Final dataset formed has audio feature and artist data for 9,758 songs out of which 3,796 were billboard hits and 5,962 non-hits with 23 features. We drop the non-numeric, non-categorical features and have 16 features post that.

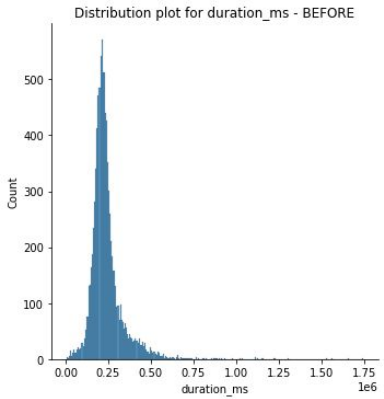
Data preprocessing and Analysis

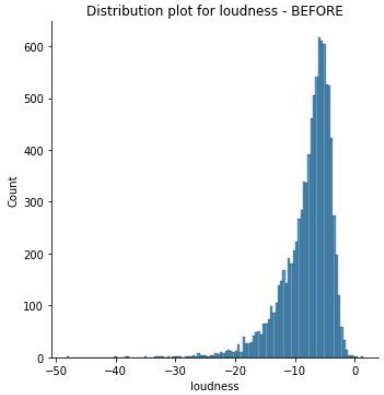
- First checked for missing values (NaN, NULL values). None were found as we handled it during extraction.
- Handling skewness and standardization :
  - Skewness was observed in the data.
  - Yeo-Johnson power transform was used to fix same. Along with it data was standardized.
  - Resultant feature distribution then approximated normal distribution (zero mean, unit variance).

$$x_{new} = \frac{x - \mu}{\sigma}$$

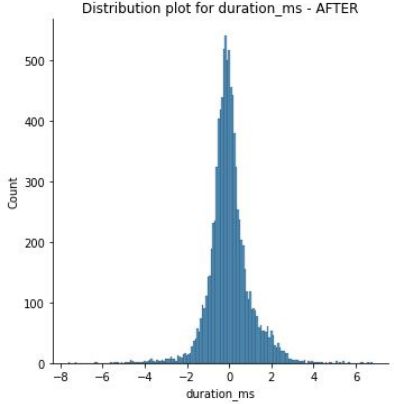
Skewness Plots

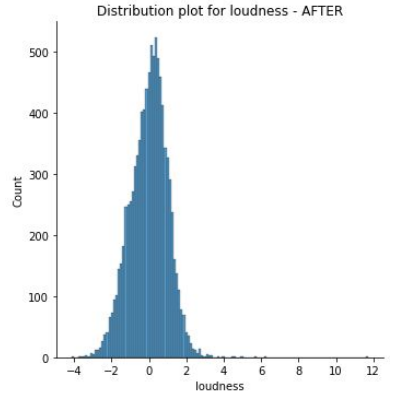
Before :





After :



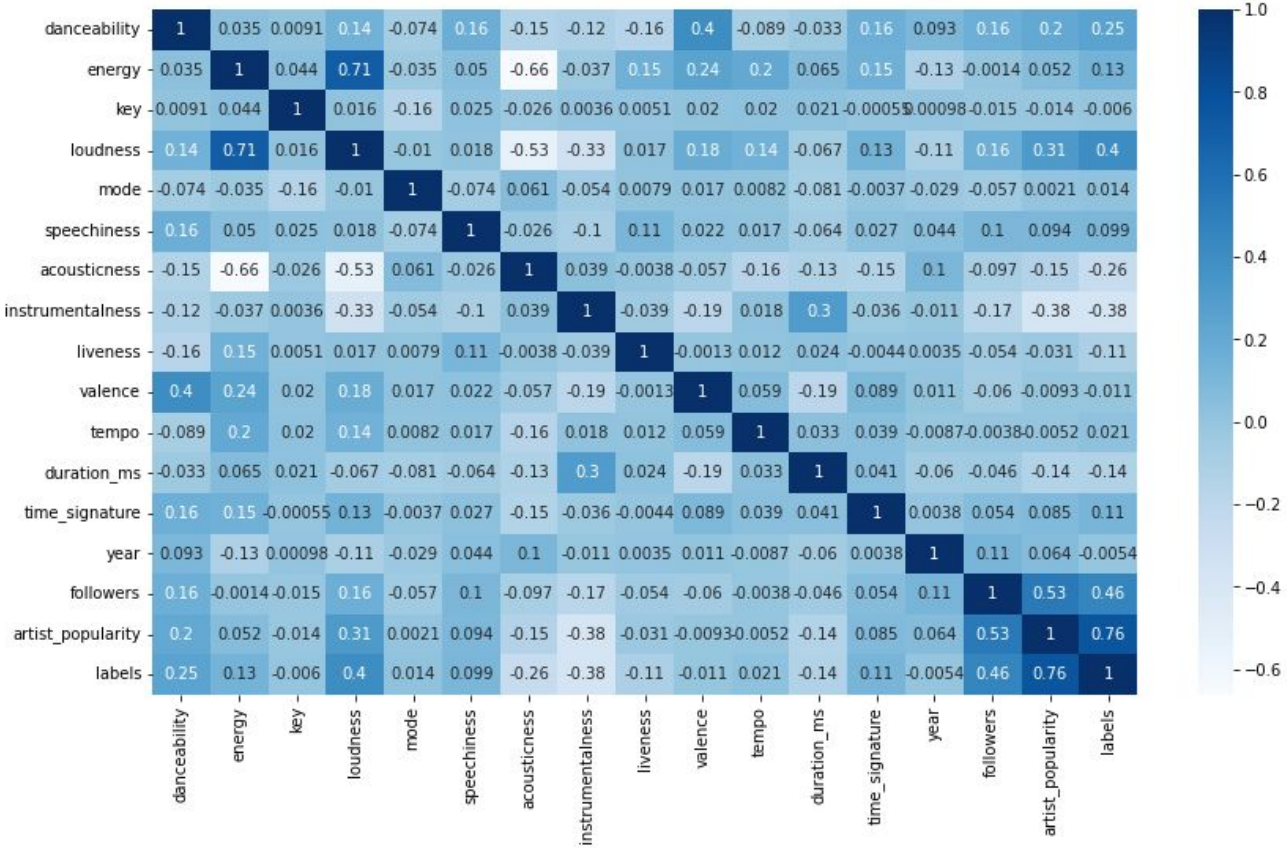


Note that the relative order of feature values remain the same since Yeo-Johnson is used to create a monotonic transformation

Correlation and feature selection

- Correlation : Indicates about how close two variables are to having a linear relationship.
- We plot a correlation matrix between the features and observe measure of correlation between features.
- Features with high correlation have similar effects on the dependent variable (prediction output). So we can drop some dependent features to make the model simple.

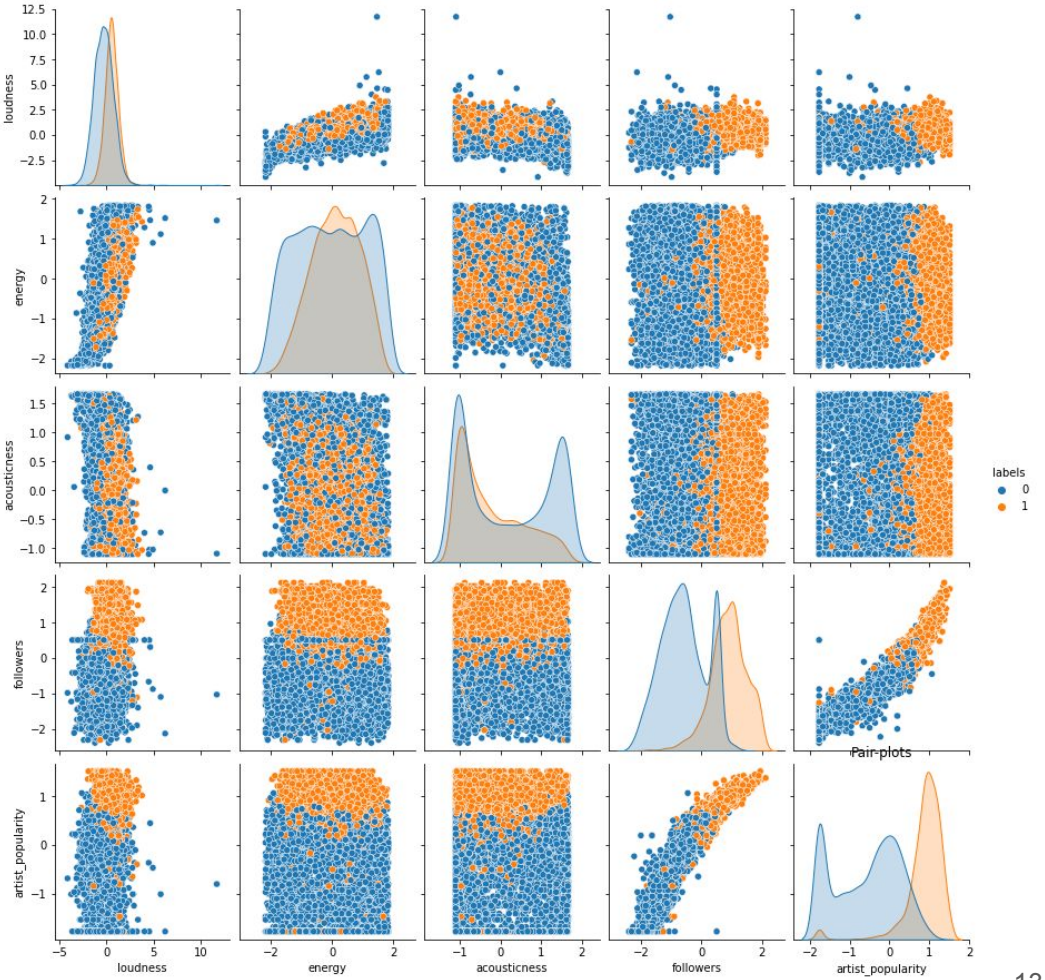
Correlation Matrix



- High positive correlation between energy and loudness
- High negative correlation between energy and acousticness
- High correlation between artist popularity and number of followers

Use of pairplots

- Visualize distribution of single variables and relationship between two variables through pairplot.
- Energy increases approximately linearly with loudness. Same pattern for artist popularity & followers.
- We chose to drop two features : **energy** and **followers**



Handling Outliers

- Outlier : A data point that is distant from other data points i.e- does not follow the general pattern of data.
- We need to remove outliers as they can cause problems related to model training and fitting as they can corrupt a dataset.
- Z-score : Intuitively how many standard deviations away given data is from mean. Same as standardized value.
- As data is standardized, we use Z-score to remove outliers. We use *threshold value* as 2.6.

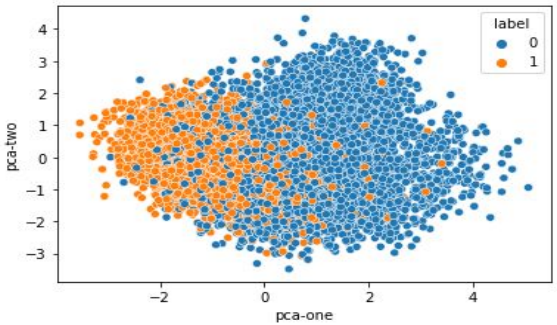
Final dataset : 8,995 songs : 3,689 hits (label 1) [41%], 5266 non-hits (label 0)[59%]

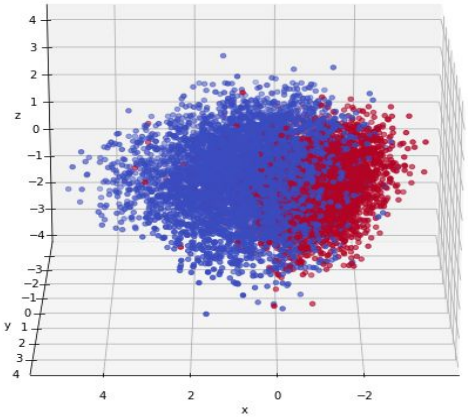
14 numerical features

Dimensionality Reduction : Visualizing PCA

This technique tries to maximise the amount of variation of the original data.

We use PCA to reduce the dimensions of the data points to three dimensions and plot the same. The explained variance ratio per component is [9.99952230e-01, 4.77703879e-05, 6.88124530e-12].



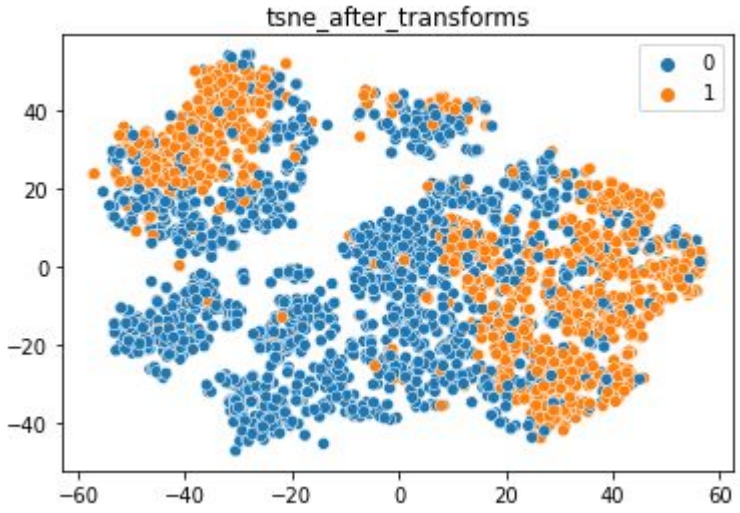


Dimensionality Reduction: Visualizing t-SNE

This technique tries to preserve the local structure of the data.

We use t-SNE to reduce the dimensions of the data points to two dimensions. Through this we can visualise higher dimensional data and get a sense of similarity between data points.

As is evident from the t-SNE plot, the data is not linearly separable.



Methodologies

Models

- We use audio features of a track to predict whether it will be a hit (featured in Billboard Hot 100) or non-hit.
- Logistic regression, decision tree, random forest and gaussian naive bayes are used
- Performance metrics used to rank models are accuracy, precision, recall and AUC score for ROC curve.
- We perform 5-fold cross-validation and grid search over hyperparameters to get the best models

Models

- **Logistic Regression -** We use logistic regression with L2 regularization and inverse of regularization constant = 1.9.
- **Decision Tree -** We use decision tree with criterion entropy and max depth 4.
- **Random Forest -** We use random forest with criterion entropy, max depth 9, number of features selected is √m (m is total number of features) and 30 estimators (this model gives best results).
- **Gaussian Naive Bayes -** We train a simple gaussian naive bayes classifier. This model gives the worst results (as expected since features are not independent).
- **SGD Classifier -** We train logistic regression with stochastic gradient descent and L1 regularization with learning rate 0.02 and regularization constant 0.1.

Models

- **SVM -** We use support vector machine to perform regression using different kernels. We boost the results using techniques like soft-margin and kernel trick.
- **AdaBoost -** It is a statistical classification algorithm that boosts the results of weak models.
- **Multilayer Perceptron -** MLP is a class of feedforward artificial neural networks. Strictly speaking, it comprises of multiple layers of perceptrons with threshold activation.
- **Convolutional Neural Network -** CNN is a class of artificial neural networks that usually used for image or image-like data.

Performance Metrics

- We use the following metrics to assess our model:
  - Accuracy = (TN + TP) / (TN + TP + FN + FP)
  - Precision = (TP) / (TP + FP)
  - Recall = (TP) / (TP + FN)
- We try to maximize precision as high precision leads to less false positives (non-hits predicted as hits). We would like to avoid spending too much money on a song that is not going to be a hit.
- ROC curve is plotted and AUC score calculated. Higher AUC means better prediction ability and performance of model

True positive : Rate $Sensitivity = \frac{TP}{TP + FN}$

False positive : Rate $FPR = \frac{FP}{TN + FP}$

Result and Analysis

High-Level Classification

Points to Note

- Our aim was to maximize accuracy and precision, high accuracy so that we have more true positives and true negatives and precision as we wanted to minimize false positives

- From the correlation heatmap - it can be said that the features are not independent from each other

- t-SNE plots show that the data is not linearly separable.

Results and Analysis

- We trained our model using 5-Fold-Cross Validation while performing a grid search on hyperparameters. The best model was found by grid search by using the precision as the scoring metric as we wanted to minimize the number of false positives (‘non-hit’ labeled as ‘hit’).
- Using the best model, accuracy, precision, recall and the F1 score was calculated for each of the classification models and we got the results as shown in the below table . (Best model is in bold)

Model	Accuracy	Precision	Recall	F1
LR(BGD)	0.9270	0.8999	0.9259	0.9127
DT	0.9218	0.9341	0.8717	0.9018
RF	0.9397	0.9436	0.9078	0.9254
GNB	0.8604	0.7751	0.9313	0.8461
LR(SGD)	0.9084	0.8708	0.9132	0.8915
ADA	0.9142	0.8810	0.9096	0.8951
MLP	**0.9720**	**0.9549**	**0.9765**	**0.9656**
SVM(LIN)	0.9157	0.8841	0.9096	0.8967

Table 1. Performance evaluation of classification models

Results and Analysis

- MLP (Multi-Layer Perceptron) gives us the best results as it gives extremely good accuracy as well as precision.

  This is not a surprise as Artificial Neural Networks are very good function approximators

- Since the data is not linearly separable we see Decision Trees performing very well on the data. Now as Random Forests are an ensemble model on Decision Trees, it combines the outputs of many Decision Trees thus performing better than a single Decision Tree.

Results and Analysis

- Among SVMs we found linear SVM to give the best results.

- Linear SVM provided better results than Adaboost, Logistic Regression and also Gaussian Naive Bayes.

- SVM does well as it tries find an optimal hyperplane so that it can classify the unseen data better.

Results and Analysis

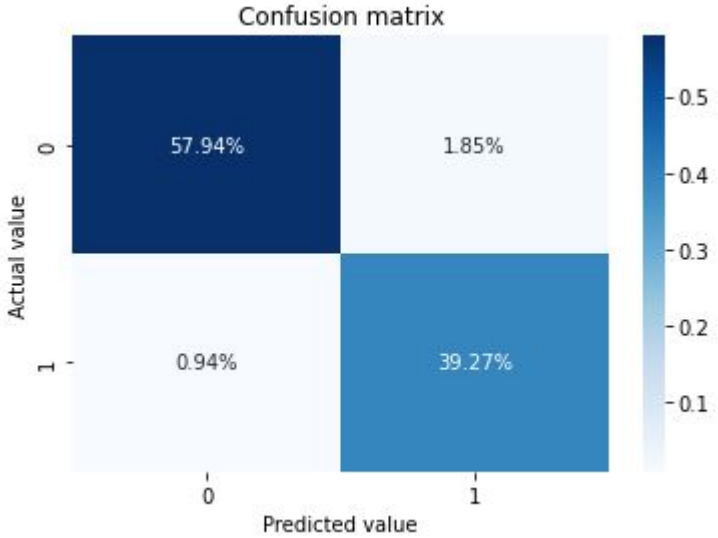
- We see a relatively poor performance by Gaussian Naive Bayes.

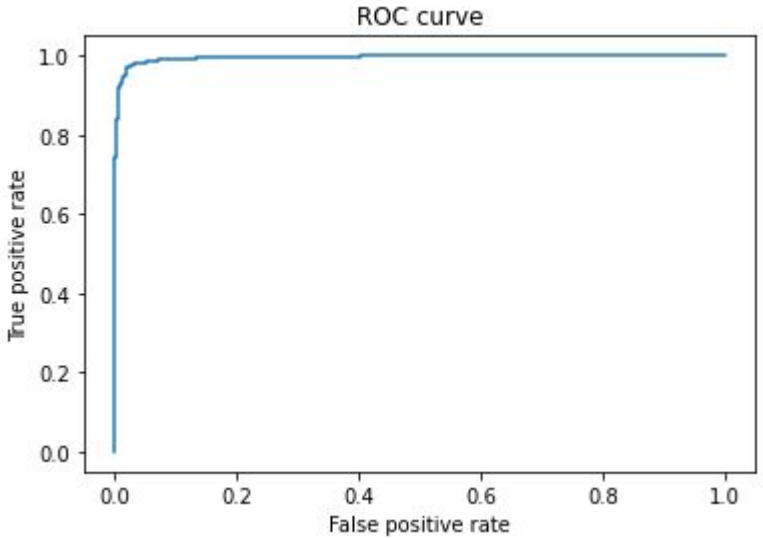
  This is because the Naive Bayes algorithm assumes the features to be completely independent of each other, however this is not the case in our data (clearly seen in the correlation heatmap).

- We also see a poor performance by logistic regression when compared with Decision Trees and Random Forest.

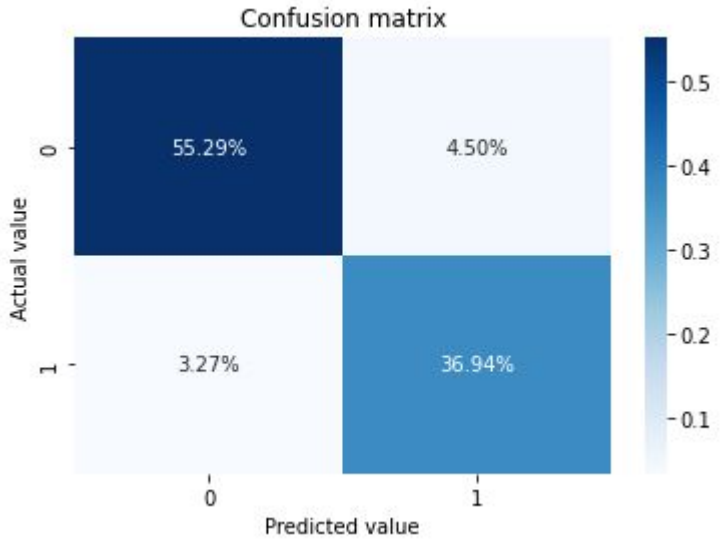
  This is because logistic regression assumes the data to be linearly separable, however this is not the case in out data (clearly seen in the t-SNE plots).

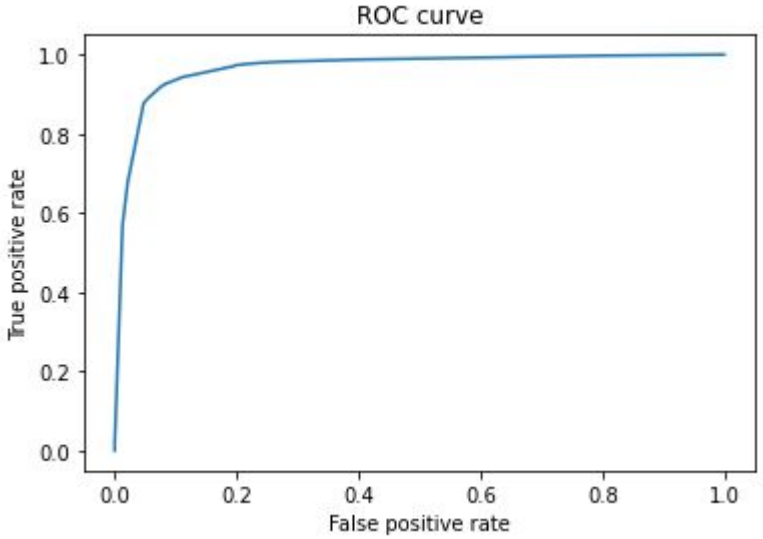
MLP Classifier



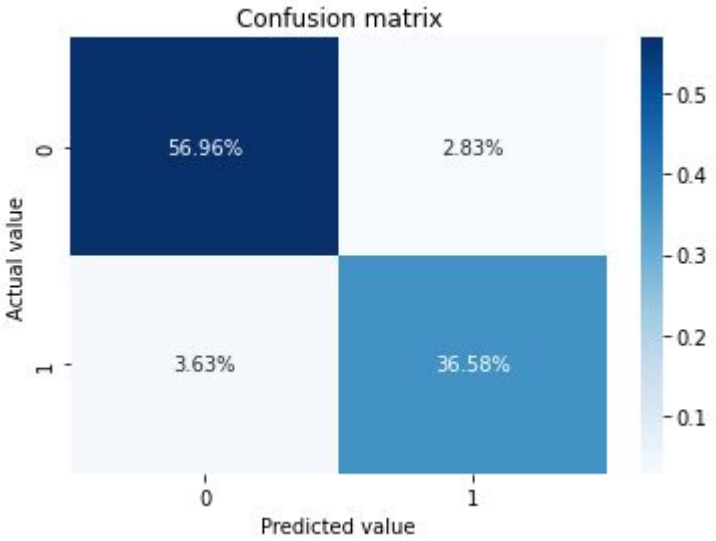


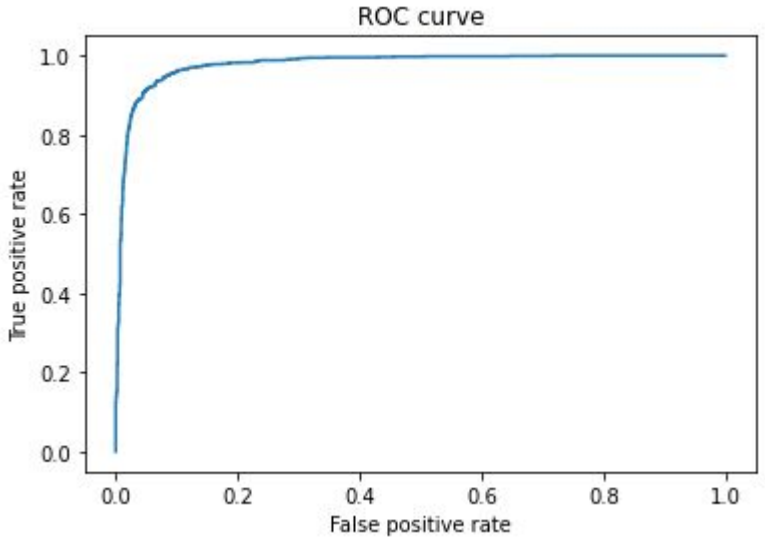
Decision Tree Plots



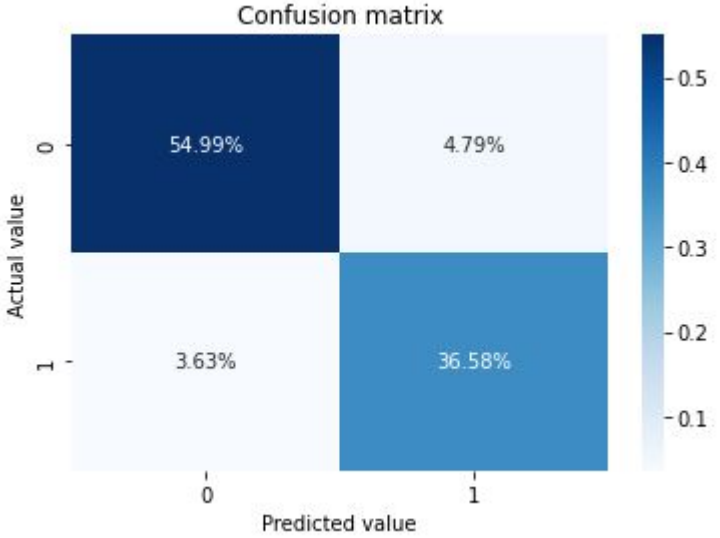


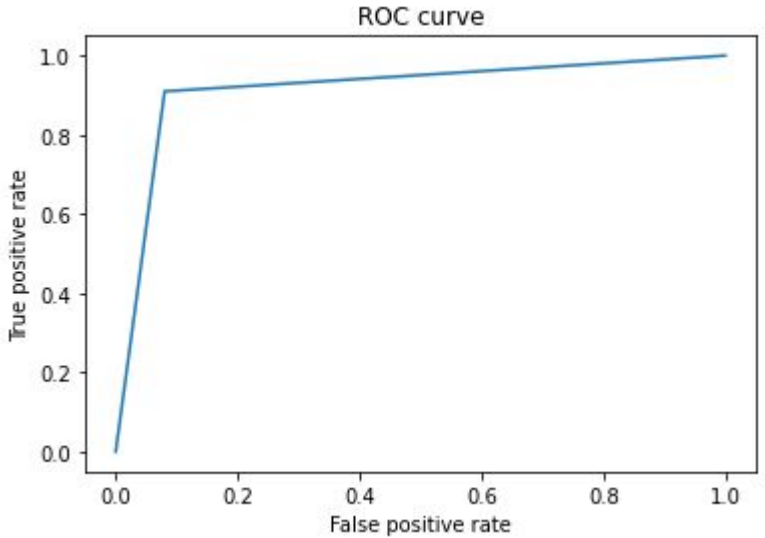
Random Forest Plots



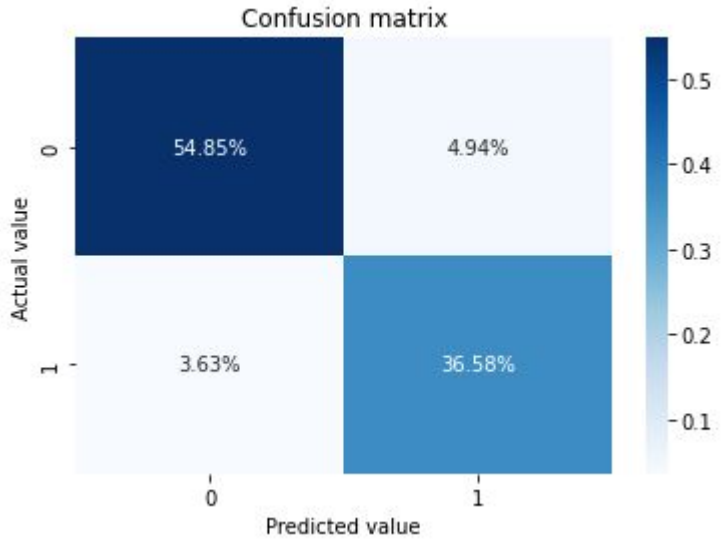


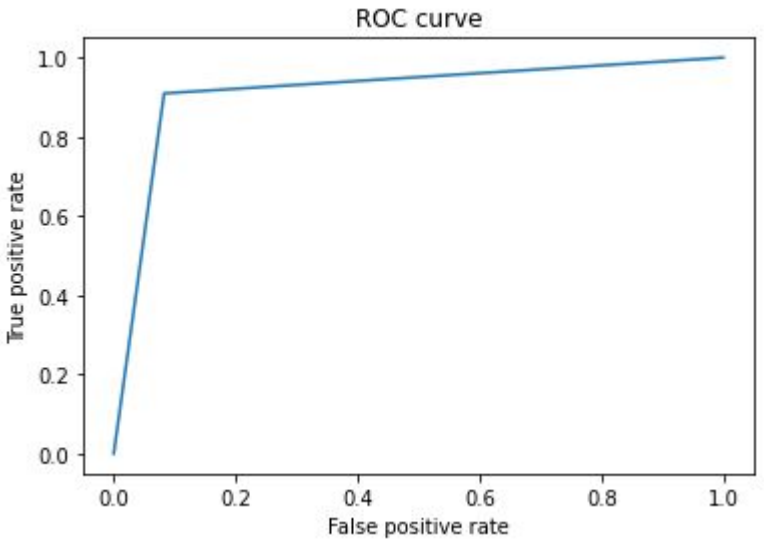
SVM Linear


Confusion matrix
54.99%
4.79%
3.63%
36.58%
Actual value
0
1
Predicted value
0
1
0.5
0.4
0.3
0.2
0.1

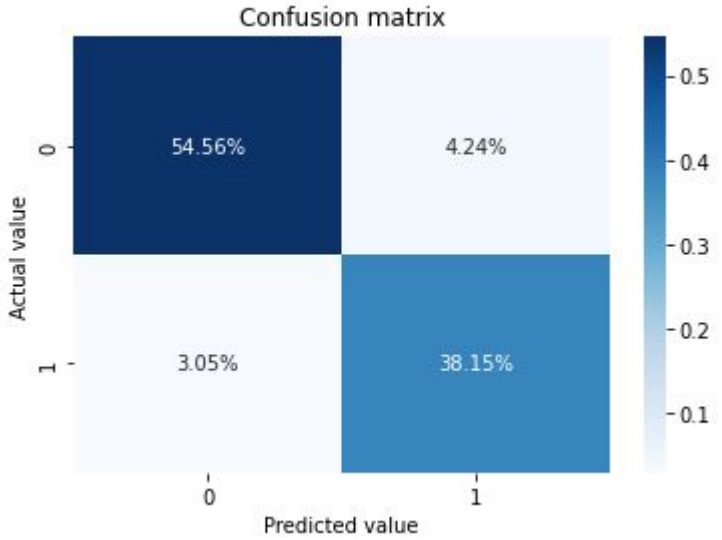

ROC curve
True positive rate
False positive rate
0.0
0.2
0.4
0.6
0.8
1.0

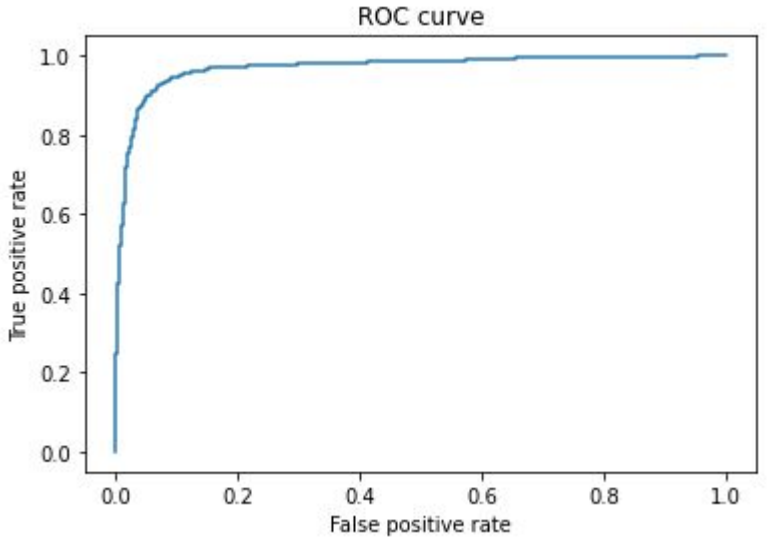
Adaboost


Confusion matrix
54.85%
4.94%
3.63%
36.58%
Actual value
0
1
Predicted value
0
1
0.5
0.4
0.3
0.2
0.1

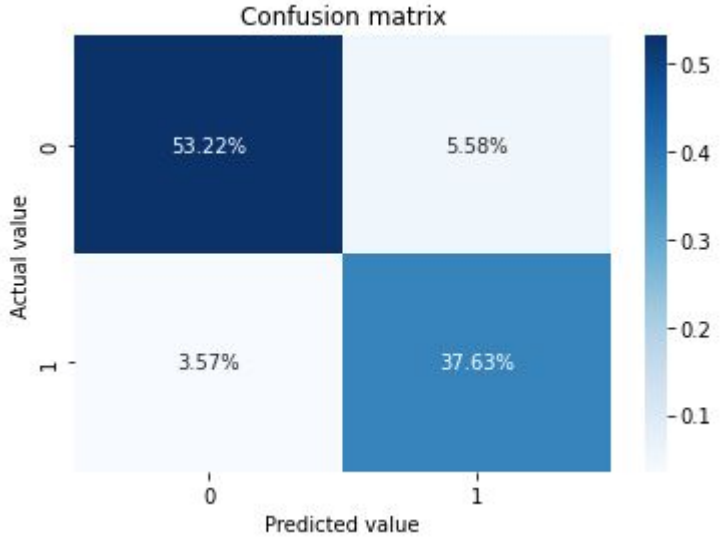

ROC curve
True positive rate
False positive rate
0.0
0.2
0.4
0.6
0.8
1.0

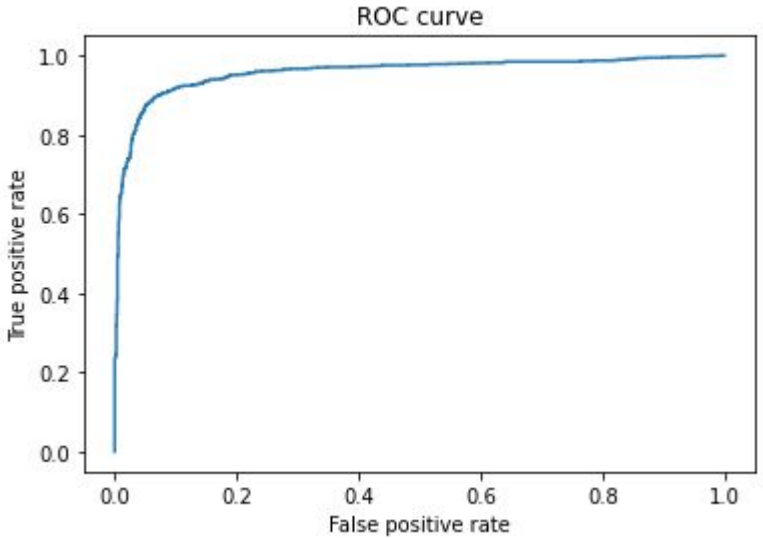
Logistic Regression (BGD) Plots



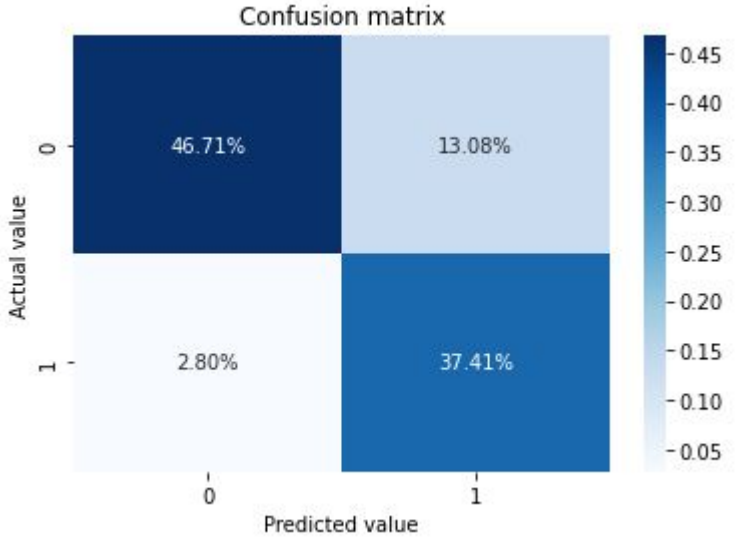


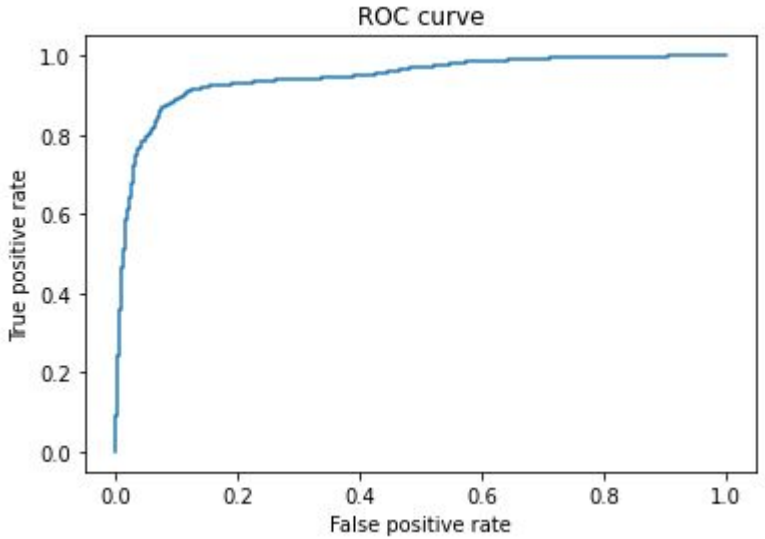
SGD Classifier Plots

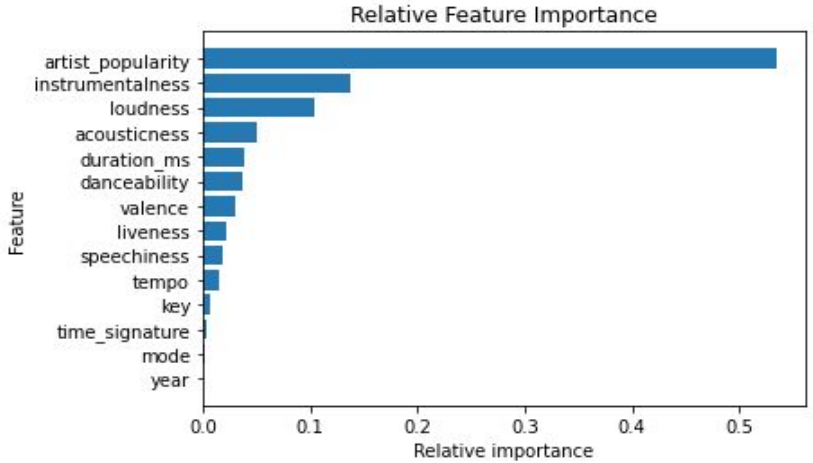




Gaussian Naive Bayes Plots







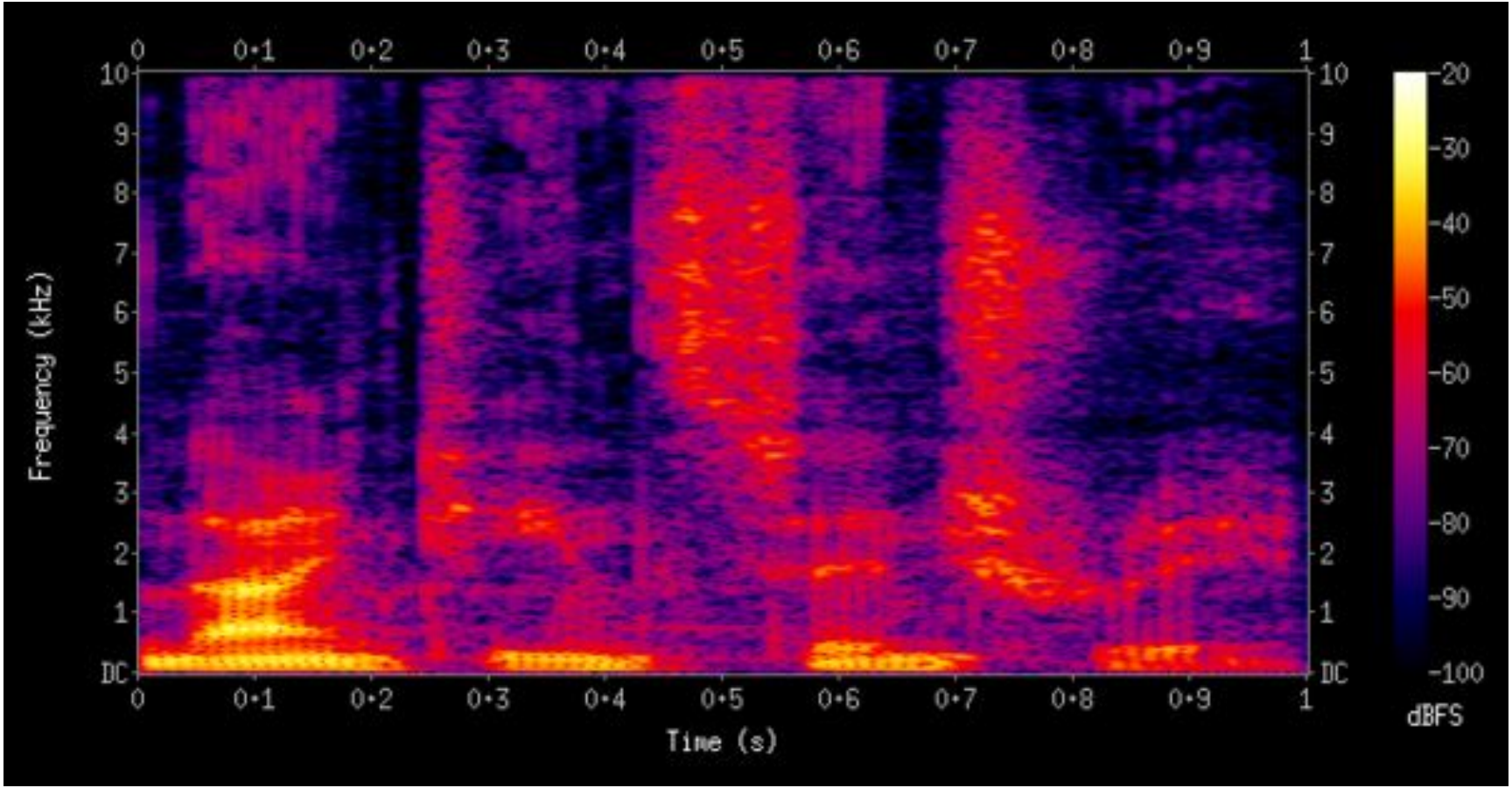
The above plot of relative feature importance in the Random Forest Classifier.

- As expected, artist popularity has the highest relative feature importance and dominates by a huge margin.
- This is justifiable because we expect artist popularity to be based on previous performance of the artist on similar metrics.
- As hit songs are equally distributed throughout the year, Year feature is of low importance.
- The plot also gives a good explanation to the correlation heatmap of the features with labels.

Low-Level Classification

Data

- Out of the 9758 songs, audio data of 7408 songs was available.
- Due to computational limitations, only 10 seconds of each song is sampled with sampling rate 44160.
- Size of each data sample is now 331264
- We do a 80:20 train test split.
- The audio waveform data is used to generate the spectrogram which is a visual representation of the spectrum of frequencies of an audio signal.

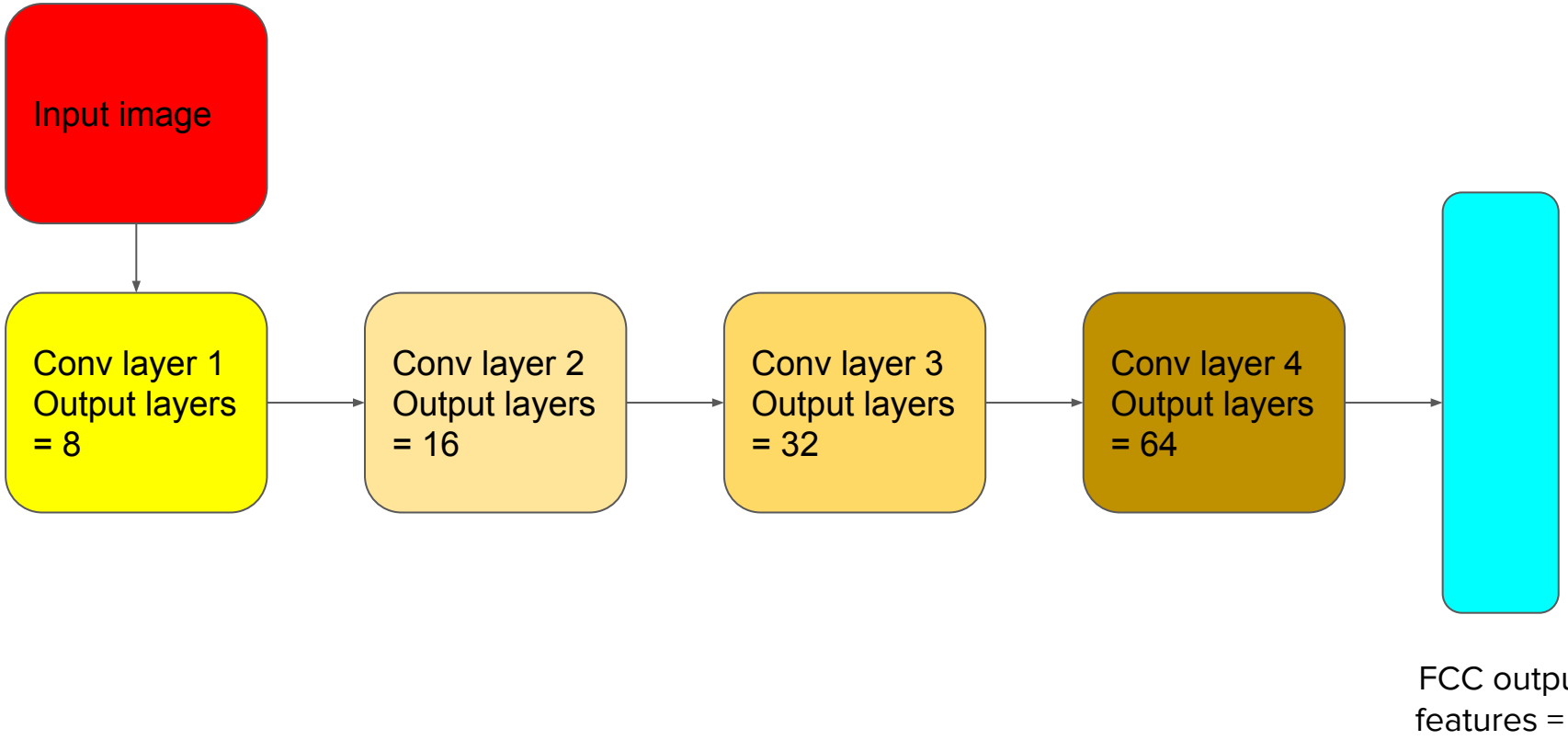


Source: https://en.wikipedia.org/wiki/Spectrogram

Example of a spectrogram

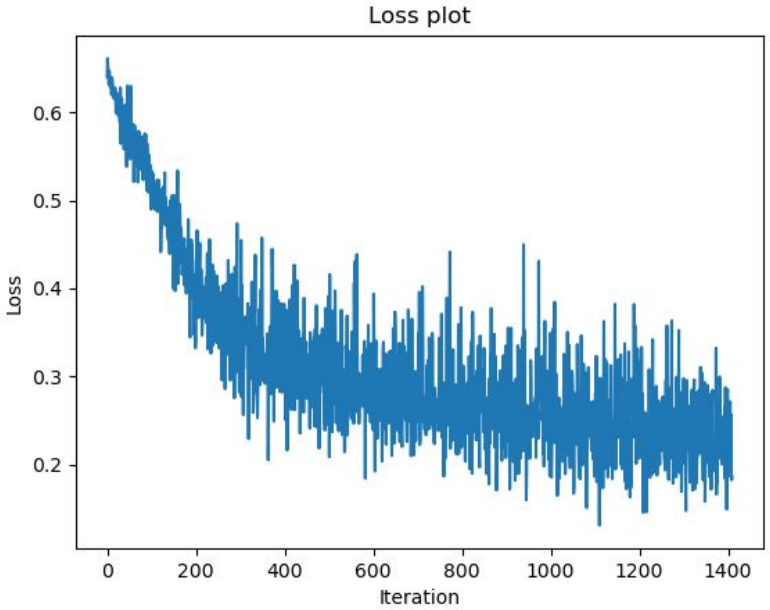
Model Description

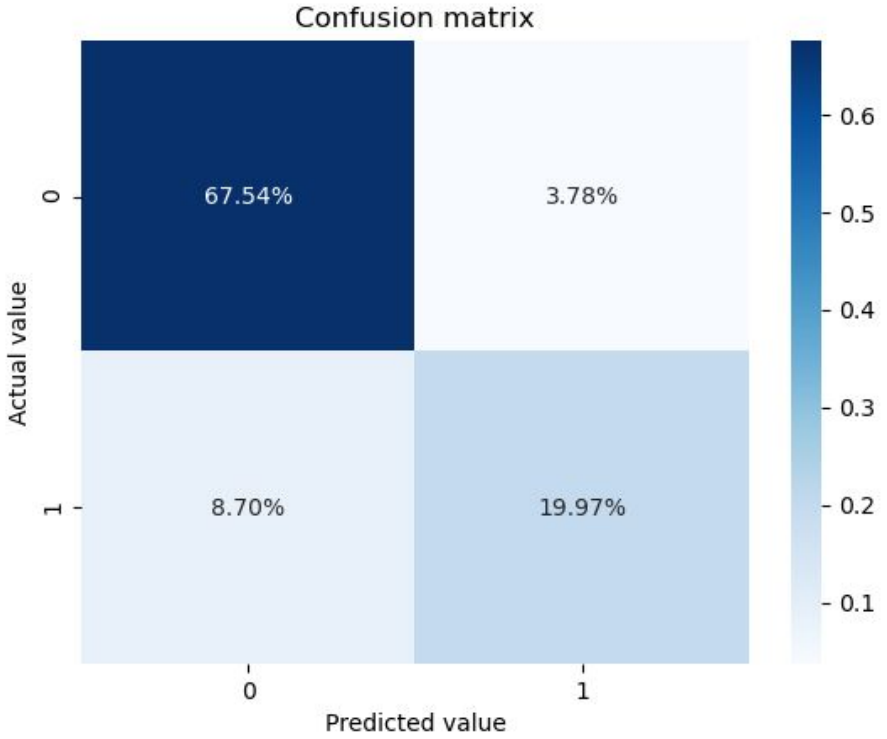
- The model consists of 4 convolutional layers and 1 fully-connected layer.
- The loss function used is cross entropy loss (binary cross entropy loss since there are only 2 classes).
- It is trained for 30 epochs with batch size 128.
- The initial learning rate is 0.001. Adam optimizer is used for gradient descent.



Model Architecture

Plots


Loss plot
Loss
Iteration
0.6
0.5
0.4
0.3
0.2
0
200
400
600
800
1000
1200
1400


Confusion matrix
Actual value
Predicted value
67.54%
3.78%
8.70%
19.97%
0
1
0
1
0.6
0.5
0.4
0.3
0.2
0.1

Results and Analysis

- Accuracy = 0.8751
- Precision = 0.8409
- Recall = 0.6964
- F1 score = 0.7619

Results and Analysis

- Even though the model performs only marginally better than gaussian naive bayes, it is important to note that we only use the raw audio data.
- No bias caused due to high level features like artist popularity and number of followers.
- A less known artist can release a potentially hit song.

Conclusion

- The project gave us a good experience in extracting data using APIs
- We learnt techniques by which we can visualize data and analyse it
- We also learnt about the methods used to perform EDA and data preprocessing.
- We learnt how to use classification models like logistic regression, gaussian naive bayes, decision trees, random forest.
- We got to learn the ways by which we can analyse the performance of our model using using various metrics like accuracy, precision, recall, F1 score and visualise the usefulness of the model using ROC curves

Future Work

Future Work

- The low-level analysis done can be improved by making the network deeper (which will require more computational resources).
- Also, combining high-level and low-level features can give even better results.
- Currently, we only use spectrogram data for low-level analysis. There are other features that can be extracted from audio and can be used for classification models

Member Contribution

Member Contribution

1. Parth Chhabra: Dataset Extraction (Spotify Data), Low Level Feature Extraction and Analysis, preprocessing, analysis-Standardization, PCA, logistic regression(SGD Classifier), decision tree and analysis,SVM models, CNN, grid search, performance analysis, Report writing.

2. Samyak Jain: Dataset Extraction (Billboard data), Low Level Feature Extraction and Analysis, preprocessing,analysis-Correlation, PCA, logistic regression(BGD),SVM models, CNN, decision tree and analysis, gridsearch, performance analysis, Report writing.

3. Sarthak Johari: Dataset Extraction (MSD), Low Level Feature Extraction and Analysis, Audio Data Extraction, preprocessing, analysis-outlier handling, t-SNE,Random forest, naive bayes and analysis, Adaboost, MLP Classifier, grid search, performance analysis, Report writing.

4. Yash Mathne:Data Extraction (MSD Subset) and Analysis-dimensionality, Low Level Feature Extraction and Analysis, Audio Data Extraction, Literature Review, Random Forest, naive bayes and analysis, Adaboost, MLP Classifier, grid search, performance analysis, Report writing

Thank You