Project Report

Saurabh Vaidya

Email- vaidya.saur@northeastern.edu

Ojas Patwardhan

Email- patwardhan.o@husky.neu.edu

Virender Singh

Email- singh.vir@husky.neu.edu

March 2020

1 Title

Predicting a hit song using audio features of songs and finding importance of features and optimum number of features required.

2 Objective and significance

- Companies working in the field of music technologies like The Echo Nest, ChartMetric, and Next Big Sound have been using data analytics to help artists and labels predict and track a song's success for almost a decade. This problem is widely known as Hit song Science in the domain of Music Information Retrieval. Our objective is to determine whether songs can be classified purely based on the musical audio features of each song. Audio features could be scores for instrumentalness, danceability, speechiness, acousticness and more. We would also determine how many features are really important to look at when making those decisions. Further, when a particular song is labelled as a hit song, what distinct features and their properties influence those decisions. We would also like to find if there are any differences in the output of our models if the datasets are used from different time periods.
- Our motivation is driven by questions like how important is music these days in a hit song and how much role does absence of an artist and lyrics play in its predictability. Ability to figure out how much the audio features of a song play a role in a song's popularity among masses can help lot of music producers and artists gain insights. After all if all music today sounds the same and hit songs are

strongly influenced by artist labels, then it would be pretty disappointing for the world of music and the art itself. Such a model, if it fails to predict hit songs based on audio or if it succeeds, can have major impact in speculating how the world of music would evolve. Failure to classify would be a sign that audio features have more or less been the same for all songs and successful classification would help in knowing which audio features influence popularity.

- We want to try with 2 datasets, one which has songs from recent 4 years while the other is span across 6-7 decades. We believe hit songs change their properties with each era and thus a model trained on 7 decades worth of songs even if it performs better with given dataset, might not be helpful to predict newly released songs. Thus we want to train our main focus to be on models trained on a dataset of songs from 2017-2020.

3 Background

3.1 Important concepts and background information

- This work is based on a hypothesis that Songs that hit the top ranks share some common features among them. We attempt to find the relations between different audio features of the song and their effect on song's popularity. This concept is known as Music Information Retrieval (MIR). Various attributes like lyrics of the songs, danceability, tones etc. can be taken into contribution. Some custom made features like Singer's popularity or Music director's past tracks can be also considered. However our focus would be solely on using audio features to predict hit songs.

- This whole work can be divided into a three block pipeline. The first is called feature learning which will rely on techniques like visualization and clustering to understand the data better. The second step is classification which given a song can predict whether the song will be ranked in top ten or not. Finally, we then draw insights into feature importances and distribution of features in our predicted hit songs and true hit songs.

- We created two different datasets first one spans from 2017-2020 and second one spans from 1964-2018. We considered a song to be hit if it was featured in Spotify top 30 in first dataset and top 100 in second dataset.

- Our data set was imbalanced because we had a lot of negative points(non hit songs) as compared to positive ones(hit songs). Hence, we decided to give each class a weight. We used this approach in all our models and we gave the minority class a higher weight as compared to the majority class.

- The next challenge that we faced was evaluating the imbalanced dataset. Since we had more negative samples than positive ones, we can not report only accuracy for evaluating our models. Thus we resort to methods such as area under ROC curve and precision values. In addition to this, our dataset was already relatively small which made using accuracy as an evaluation strategy all the more unsensible.

- The methods which we used for these purposes are Neural Networks, Logistic Regression, Random Forest and SVM.

- SVM is a supervised learning method that looks at data and sorts it into one of two categories. It is trained with a series of data already classified into two categories, building the model as it is initially trained. The task of an SVM algorithm is to determine which category a new data point belongs to.
- The random forest is a model made up of many decision trees. These decision trees operate as an ensemble. Each of the decision trees then predict the class label and the majority prediction is considered the models prediction. Thus, we also implemented ensemble learning.
- Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable. The logistic or sigmoid function is given by the formula $P(Y = 1|x, w) = (1 + e^{-t})^{-1}$.
- Neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input, so the network generates the best possible result without needing to redesign the output criteria. A decision tree is a largely used non-parametric effective machine learning modeling technique for regression and classification problems. To find solutions a decision tree makes sequential, hierarchical decision about the outcomes variable based on the predictor data.

3.2 Literature review

- In [1] authors have collected a dataset of approximately 4000 hit and non hit songs and extracted each song audio features using spotify web API. They predicted which songs will become top 100 billboard hot songs with approximately 75% accuracy. The most successful algorithms were Logistic Regression and a neural network with one hidden layer. Data was collected using [3] which is a research project stemming from a New York Times article that used the Spotify audio features to illustrate the similarity of summer songs. Yu-et al[4] in their paper do it hit song prediction exclusively for pop music. Authors model a ranking system for pop music and use a convolutional neural network for the same. They train a multi-objective CNN model with Euclidean loss and pairwise ranking loss to learn from audio the relative ranking relations among songs. Researchers [5] have used K-means clustering to find groupings of the songs that are hit and those which are not. They make use of feature learning to learn the features from the data itself. This technique is very effective even with the simple clustering algorithms such as K-means. In [2] authors address the problem of dance hit song prediction to understand what actually makes a song hit. The database has songs spanning from the period 1985-2013. They could successfully design a model that could predict whether the song will be in the top 10 or not.
- Another group of researchers used Support Vector Machines (SVM) to predict top 10 Dance Hits [4]. By narrowing the scope of the study to only dance music, researchers were able to present a more focused work. Another study attempted to classify songs based on lyric content [6]. While they successfully classified many hits, they also returned many false positives and concluded that analyzing lyrics is an ineffective approach to this problem. The work done in [7] tries to learn a classifier that predicts if the author would like the song or not. The positive dataset is all the songs from the author's spotify playlist and negative data is a playlist of the songs the author doesn't like. Several classifications algorithms are implemented and its concluded that decision trees gave the highest accuracy for the given data. Later

the author builds a logistic regression model to find out the coefficients of features with highest values to reason why the author likes the songs. Herremans et al. (2014)[8] describe their hit song predictor that uses various audio features and temporal features. They train their models of decision trees, Naive Bayes, SVM, logistic regression and rules based classifier with a slightly imbalanced distribution of positive and negative samples. The ROC AUC is the highest for logistic regression.

3.3 If there exists previous work on the problem, describe what makes your work distinct or particularly interesting

- Work done in [1] closely resembles what we are trying to do. They also predict a song as hit/non-hit and then predict the rank as well. However, our approach differs in one major area: they consider artist's recognizability as one of the feature and this is exactly what we want to omit. Our idea of trying to predict hit songs is based on musicality of the songs only. We strongly distance ourselves from using any non-audio related features as our fundamental question remains to find importance of music in a song. Secondly, their dataset spans across 3 decades yet only has a size of 4000 songs which could suggest some selection bias. Songs from across 3 decades would lead to some higher separability amongst class labels due to more variance in features as compared to a timespan of 4 years. Furthermore, our approach would aim to reduce the number of features they have used to as less as possible while keeping precision relatively similar. We would also simulate their dataset too and try to achieve a better or equivalent performance while tackling this problem solely from a music or audio standpoint.

- Work in [2] predicts only dance hit songs and [4] only work for hit songs in pop genre. Our approch differs in that we work for all genres. [6] use lyric content which is again a non musical feature and thus different from our idealogy of music-only features. [7] has a problem statement tailored to user's personal choice of a song classified as liked/disliked by a user. [8] also work on dance hit songs and also use temporal features in addition to audio features.

4 Methods

4.1 Describe your data and how you obtained it

- Our data comprises of songs as datapoints and their corresponding audio features as features. For the purpose of this project, we used two datasets, one was top 30 songs from 2017 to 2020 which consisted of approximately 1000 positive and 6000 negative samples. The other dataset that we used was the top 100 songs dataset which had around 1500 postive and 7700 negative samples. We got all those songs from the million song dataset [14] which consists of songs from 1964 to 2018.

- For the top 30 songs dataset, we got the positive samples from Spotify Charts using Spotipy [13], which is a Python library for Spotify. We got the id's of all the songs which were in top 200 from 2017 to 2020 for each week. From these songs, we selected the top 30 for each week and created our positive samples for the top 30 songs dataset.

- Generating the dataset of negative examples was a little more complex and we had to perform web scraping to gather all samples. We scraped Wikipedia to get the names of all the songs for each year. From the names of these songs, we used Spotipy to get the corresponding id's. The JSON structure of the object returned was complex so we had to use regular expressions to parse the JSON object returned to get the id of the song. To generate our negative samples, we then removed all the id's which were also present in the positive samples that we got. This gave us our negative samples dataset. After this we proceeded to get audio features for both our negative and positive samples.
- The audio features that we got from the API were danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, duration_ms, time_signature, release year, genre, explicit.
- From here on, we will refer to the top 30 songs dataset as dataset 1 and top 100 songs dataset as dataset 2.

4.2 Describe your methodology: give flowcharts, diagrams, pseudocode or formulas where appropriate

- Before we can send this data to our learning models, there is a lot of data processing and feature engineering that needs to be done.
- The data was z score normalized. In this technique every column of the data is normalized using the mean and standard deviation of that particular column and can be expressed in the following manner :

$$x = \frac{x - \mu}{\sigma}, \tag{1}$$

  where μ is the mean of the column and σ is the standard deviation of the column.
- The z score normalization technique helps in normalizing the variance in the data. The variance in the data features in general can be quite high and thus can affect the proper training by learning algorithms.
- After we normalize each feature, we plot distributions of each feature for both labels in a single plot. This helps us to get a feel of how each feature is distributed with respect to both labels. Understanding this can help in feature selection. Results of those plots will be shown in Results section. For both datasets, models are trained with all features and with selected subset of features to confirm redundant features. After a final set of features are selected we can build a pipeline for the data.
- As seen in the diagram, the datasets first go through preprocessing phase where we perform z-score normalization on all features. Due to imbalanced label dataset, we compute class weights for each label which will be passed to our models to penalize misclassifications with a cost associated with each label.
- We split the dataset into training set and the test set in a stratified fashion. Then we train a variety of models such as RandomForest, SVM, Neural Networks and Logistic Regression on the training set.

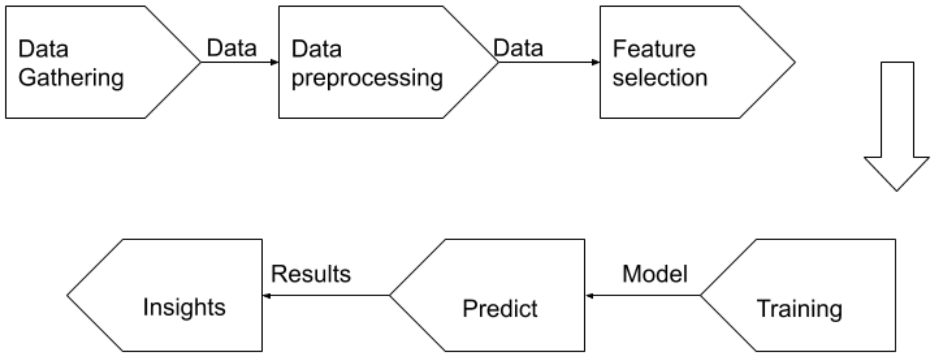


Figure 1: Flow diagram of project

- We perform 10 fold cross validation and then choose a final model which performs best in cross validation. The final model is trained using the entire train data. Logistic regression, SVM are trained with Bagging.
- After training, the data goes into the next phase of pipeline which is evaluation. We use bootstrapping to provide confidence intervals on our evaluation metrics. After training in each of the 10 folds, we evaluate area under roc curve and precision sample means and their standard errors. On the final trained model, we pass the test data and evaluate the same metrics and compare our models using those metrics.
- We also compute recall but don't use it for evaluation. We use it only as threshold for each model to be considered.
- As will be seen in the results below, for dataset 1, random forest model outperforms all other models. We then use this model to get feature importance scores. The probability scores of the model can also be used to rank our predictions which can be thought of as analogous to rank prediction
- As an additional task to verify our results with more spread out data in time, we train models with dataset 2. This dataset consists of songs from 1964 to 2018. This dataset goes through the same steps as above.

4.3 Evaluation strategy

- We used precision and AUC ROC to evaluate the performance of our models on the top 100 and top 30 song datasets.
- We compute recall as a threshold criteria to consider learning algorithms. We do not go ahead with models who have a recall of less than 0.2 even if they have high precision since our predicted positives could be too low.
- Since our dataset was imbalanced, we could not use classification accuracy for evaluation purposes as it would have resulted in incorrect inferences.
- The primary reason for using precision as one of our evaluation strategies was that we needed to know how many songs we predicted as positive were indeed true positive.
- For the same reason, we also used AUC ROC as a performance metric because it gave us a plot of true positive rate vs false positive rate. Area under the curve helped us to know what's the probability that a randomly chosen positive sample had a higher prediction score than a randomly chosen negative sample.
- We compute training performance estimation using cross validation and use bootstrapping on final trained models for confidence intervals.

5 Results

For all of the learning algorithms, we designed following experiments for both datasets:

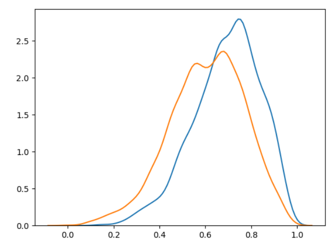
- Training with all the features present in the data
  Here we keep all the features of the dataset and try to learn on that basis. The major assumption is that it contains all of the information that dataset can provide and hence is a full knowledge training. However, this is not always a good thing when dimensions are too many.
- Training on dimensionality reduced training data.
  Here, by analyzing all the features and ranking them based on their importance, we select features with maximum information and variance which is the major idea of dimensionality reduction technique. The major assumption here is that by selecting all the important features, the model can train better without suffering from the curse of dimensionality.

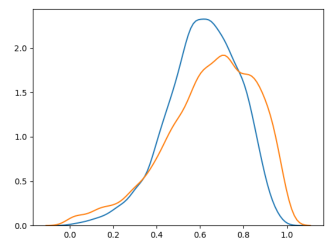
The evaluation was done using two methods :

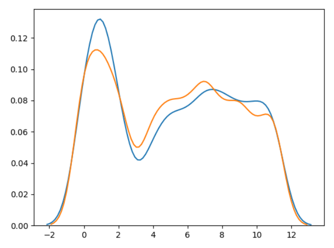
- 10 fold cross validation : Here 9 parts of the dataset were kept for training and the model was tested on the 10th part. This was done ten times for different training and testing parts and finally results were averaged for each fold.
- Full features were trained in cross validation and as we see our metrics improved when we reduced the features.
- Bootstrapping : Here we sample the test set from the original test dataset with replacement This is done for 100 iterations and sample mean and standard error for each metric are found out. This is carried out only on the reduced features' test data with the final trained model.

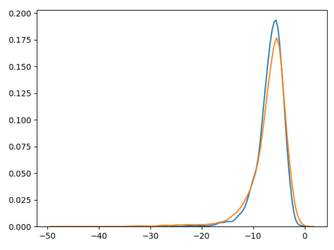
5.1 Feature selection

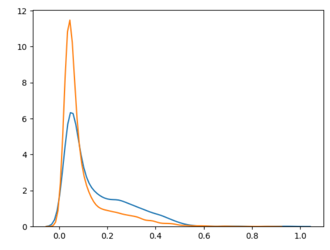
- First, we applied standard dimensionality reduction techniques like PCA but the results were not satisfactory and we got only 1-2 features with 99% variance of the data. Thereafter, we analysed the data using their distribution plots as shown in Figure 2 and Figure 3.
- Here, visually we could see which features were more important than the others. As the less important features have similar plots for both positive and negative labelled data points, we can drop these features.
- Eventually, we are only left with the features with significant difference between the positive and the negative labelled data points.
- It can be seen from figure 2 for dataset 1, 4 features seem to show some separation in their distributions. These 4 features corresponded to Danceability, Energy, Speechiness and Instrumentalness.
- Similarly as seen in figure 3 for dataset 2, 5 plots are shown out of 9 features which show some separation in their distribution. These 5 features correspond to Dance ability, energy, loudness, accousticness and instrumentalness.

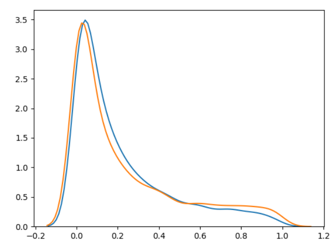

(a) Dance ability

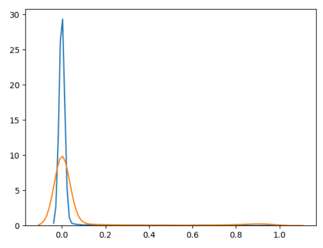

(b) Energy

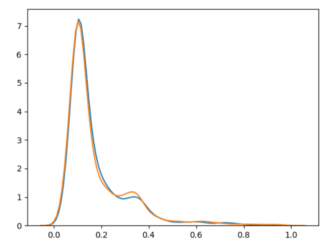

(c) Key


(d) Loudness


(e) Speechiness


(f) Acousticness


(g) Instrumentalness


(h) Liveness

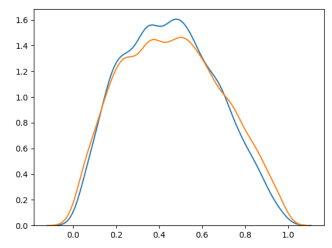
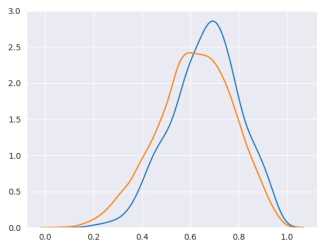
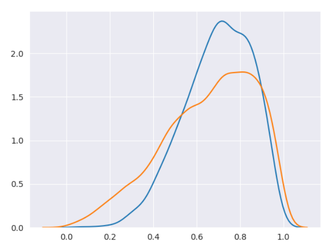
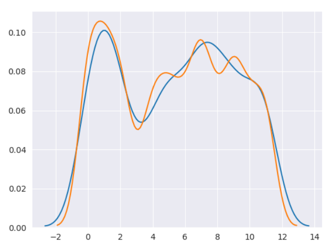
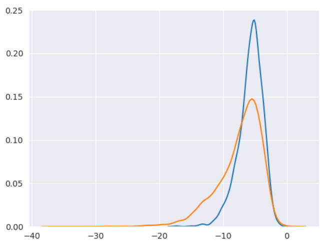

(i) Valence

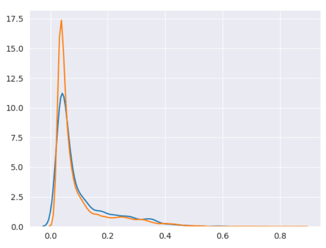
Figure 2: Feature distribution for first dataset orange: Hit song, blue: Non-hit

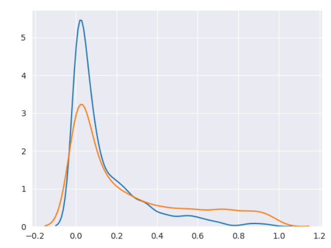

(a) Danceability

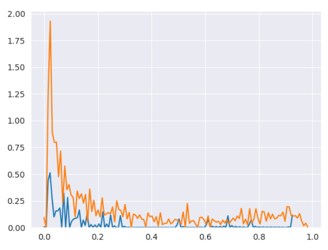

(b) Energy

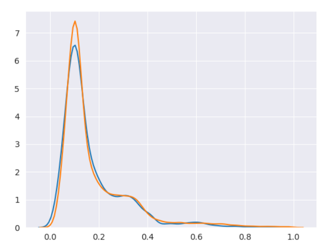

(c) Key


(d) Loudness


(e) Speechiness


(f) Acousticness


(g) Instrumentalness


(h) Liveness

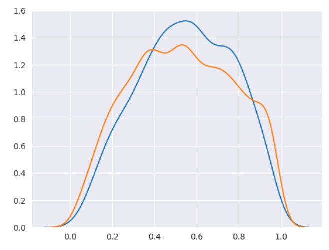

(i) Valence

Figure 3: Feature distribution for second dataset orange: Hit song, blue: Non-hit

5.2 First Dataset

For the first dataset, we have around 7000 songs from 2017-2020 and 13 features where each feature represents a particular audio feature. We train our models and estimate cross validation performance on two scenarios: all features and reduced features. We pick the best model in cross validation as our final model and then train the entire training dataset. We then evaluate test data performance with bootstrapping and report them in tables 1 and 2.

As we can see, the performance after reducing features is better than using full features. Also Random Forest does a better job than other models in precision as well as area under ROC curve.

5.3 Second Dataset

For the second dataset, we have around 10000 data points and 21 columns where each column represents a song feature and each row represents one particular song. We evaluate our result using two methods. In the first method, we train our machine learning models on the whole dataset using all the features. Results of

Table 1: Full dimensionality results with first dataset. Confidence intervals are in %

Algorithm	**Accuracy (0-100)**	**AUC (0-1)**	**Precision (0-1)**
Logistic Regression	65.2 ± 0.5	0.62 ± 1.1	0.19 ± 0.2
SVM	66.5 ± 0.7	0.66 ± 0.2	0.25 ± 0.4
Random Forest	73.7 ± 0.8	0.67 ± 0.6	0.61 ± 0.9
Neural Network	56.1 ± 0.2	0.54 ± 0.9	0.45 ± 0.01

Table 2: Reduced dimensionality results with first dataset.Confidence intervals are in %

Algorithm	**Accuracy (0-100)**	**AUC (0-1)**	**Precision (0-1)**
Logistic Regression	71.2 ± 0.2	0.75 ± 0.9	0.26 ± 0.4
SVM	74.7 ± 0.8	0.71 ± 0.3	0.25 ± 0.2
Random Forest	85.1 ± 0.9	0.82 ± 0.6	0.65 ± 0.5
Neural Network	73.2 ± 0.1	0.62 ± 0.8	0.53 ± 0.1

this experiment are shown in Table 3. Although we are getting very high accuracy for each model, this is not sufficient for evaluation of a model on the imbalanced dataset. We report model results with a recall value of more than 0.2.

Table 3: Full dimensionality results with second dataset. Confidence intervals are in %

Algorithm	**Accuracy (0-100)**	**AUC (0-1)**	**Precision (0-1)**
Logistic Regression	85.81 ± 1.33	0.59 ± 0.76	0.86 ± 1.77
SVM	88.46 ± 2.05	0.71 ± 1.04	0.83 ± 3.4
Random Forest	87.05 ± 0.81	0.70 ± 2.71	0.85 ± 1.04
Neural Network	83.97 ± 1.22	0.72 ± 0.66	0.84 ± 0.98

In the second method, we reduce the dimension of our data and select the most important features. The models are trained on these features and are evaluated based on the proposed evaluation metric. The results are show in Table 4.

5.4 Insights

- For songs from 2017-2020, we filtered our outputs to only have hit predictions. Using our trained Random Forest model, we got the decrease in gini impurity for each feature. Based on that we could derive feature importances for each feature using sklearn's method which works on averaging the decrease in gini impurity. We present those in following table:

Dance ability	**Energy**	**Speechiness**	**Instrumentalness**
0.2919306	0.29598663	0.28226125	0.12982152

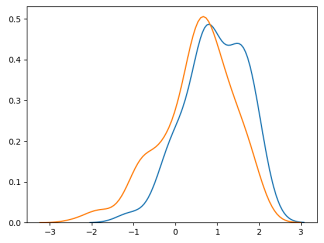
- After knowing importance of each feature, we also want to know if our model can mirror feature distribution for true positive datapoints.

Table 4: Reduced dimensionality results with second dataset. Confidence intervals are in %

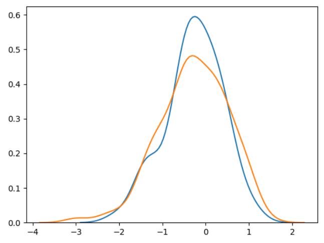
Algorithm	**Accuracy (0-100)**	**AUC (0-1)**	**Precision (0-1)**
Logistic Regression	82.51 ± 1.45	0.56 ± 0.25	0.67 ± 2.7
SVM	88.19 ± 1.54	0.71 ± 2.98	0.87 ± 1.48
Random Forest	85.59 ± 0.69	0.65 ± 1.6	0.87 ± 1.5
Neural Network	87.12 ± 1.62	0.79 ± 1.1	0.87 ± 1.06

- For these 4 features we plotted the distribution of features for predicted hit datapoints and the true hit datapoints.
- Even though random forest was the best model from our options, the mean of predicted vs true datapoints vary as seen in the table and figure 4.
- Energy and Instrumentalness have similar mean suggesting our models are able to recognize true distribution for these whereas the other two have a difference of at least 0.4 which suggests our model could not learn those feature distributions.
- Below table represents mean value for each feature for true and predicted positive samples.

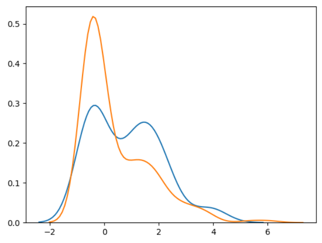
Hit Label	**Dance ability**	**Energy**	**Speechiness**	**Instrumentalness**
True hit mean	0.512	0.275	0.416	-0.223
Predicted hit mean	0.943	0.275	0.817	-0.269



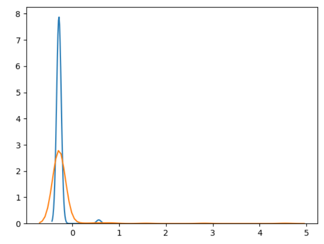
(a) Dance Ability



(b) Energy



(c) Speechiness



(d) Instrumentalness

Figure 4: True vs predicted hit song feature distributions. Orange-True, Blue- Predicted

6 Conclusions and Future Work

- For both datasets, Random Forest performed better than other models in terms of area under ROC curve and precision values.
- The modern dataset (2017-2020) is less classifiable as compared to old dataset(1964-2018). The higher accuracy on the latter could possibly be due to higher influence of music in songs in early decades from 1960s to 80s. Also the second dataset gives more variance since audio features are collected from songs across different time periods.
- Based on the distribution plots, it seems that more features have overlapping distributions in modern songs. This could be validated since songs earlier used to have different writers and thus added variance in audio features. It's a known music industry fact that, nowadays, most of the songs today are written by same music writers thereby resulting in hit and non-hit songs being almost similar in audio features.
- Random Forests give the best results on this dataset. After looking at the distributions, it can be reasoned that on such datasets with overlapping distributions and imbalanced datasets, tree based models might perform better. Since we would need to take a lot of decisions at various thresholds of our features, decision tree should work. Random forest, being an ensemble of trees constructs a better model than stand alone decision tree.
- It can be concluded, somewhat unshockingly that modern songs are strongly influenced by Dance ability, energy, speechiness and instrumentalness. If one can imagine the songs that get popular easily these days viz. hip-hop, pop, edm songs, all seem to have more dance ability, energy, speechiness(rap) and less instrumentalness which in a way validate our observations.
- While we were able to confirm our hypothesis that audio features alone can predict hit songs and find features that dominate these decisions, the classifier performance on hit song prediction is not good enough. Our top model has 0.67 precision and 0.75 AUROC on modern songs. Thus audio features alone may not be sufficient enough to build a powerful hit song predictor.
- We believe this work can be improved by splitting songs by genres and building individual models based on genres. This would provide higher accuracy in our hit song predictions. Moreover, incorporating artist names and lyrics can possibly improve prediction quality significantly when coupled with the 4 features mentioned above.
- On building models, we can experiment further, tuning parameters for certain learning models such as trying different estimators in random forests. For neural networks we can try out more architectures, activation function and hidden neuron configurations. We can also try running deep neural networks on these datasets.
- Moreover, our approach relied solely on spotify's feature calculation methods. Maybe computing audio features differently from either different sources or with the help of domain expert can improve the performance.

7 Individual tasks

- Ojas Patwardhan : I worked on generating negative samples (non-hit songs) for dataset 1. To gather the negative samples, I had to scrape Wikipedia and get the names of all the songs which were released between 2017-2020. For each year, I got approximately 3000 songs. After this, I got the id's of all these songs using Spotipy. I used regular expressions to get the id's from the JSON object returned by Spotify Web API. I then gave all of these id's to my teammate to generate the final dataset. I performed data cleaning and preprocessing on this final dataset and carried out feature analysis using PCA to reduce the number of features to be used and also to extract the features which accounted for the most variance. For PCA, I experimented using 99% retained variance. I then proceeded to train a Neural Network by implementing cross validation to partition into train and test sets. I experimented by tuning hidden layers, number of neurons, activation function, epochs and other hyperparameters. I used bootstrapping to calculate the mean and standard error. Post this, I calculated precision, recall and AUC ROC for the model. After discussing the evaluation results with my teammates, I drew inferences as stated in the conclusions part. I was also tasked with writing parts a and b of background and parts a and c of methods sections.

- Saurabh Vaidya : I worked on acquiring datapoints for positive labels. I searched spotify to find helpful playlists of hit songs. After searching I selected spotify charts as my data source for positive(hit songs) songs. Then I used spotipy for extracting top200 weekly charts from 2017 to 2020. From those charts I extracted spotify uri ids for each song. I acquired audio features for those ids and stored them. After obtaining a complete dataset and preprocessing, I worked on analysing the features to determine which features are redundant and can be removed. I worked on plotting distribution of each feature per each class label. After plotting the distributions, I tried different combinations of a subset of those features and finalized our feature set to have 4 features for first dataset. After data preprocessing, I worked on training ensemble models. I ran random forest and applied bagging to logistic regression and svm models built by my teammates. For each of my models I used cross validation and bootstrapping for performance evaluation. I computed area under ROC curve and recall and precision for my models. I worked on finding feature importance from random forest model and contributed in our team discussions for discussing conclusions and inferences that we can draw and critically reason and rationalize our results. In writing the paper, I worked on writing the objective and significance, part c of background. I also made the flow chart and wrote the methodology part b. I contributed wrote some parts of the conclusion in discussion with my teammates.

- Virender Singh : I worked on collection of second dataset through various sources like kaggle contests, spotify, wikipedia and billboard charts etc. After doing preprocessing and feature analysis, I finalized a dataset of approx 10000 songs and 21 features. I plotted feature importance plots which showed the distribution comparison of features for positive and negative samples for the second dataset. For model evaluation, I developed and designed logistic regression and SVM models. These models were run using cross validation and bootstrapping techniques. I designed two different experiments of the second data where first one is performed on the complete feature list of the data and second one is performed after dimensionality reduction technique. I applied dimensionality reduction on the second dataset to come up with reduced feature set and trained my model on this. Results of both the experiments were evaluated based on the accuracy, area under the ROC curve and the precision values. I did hyperparameter training using grid search to obtain the best parameters for the different models and finally trained the best resulting model for both experiments. Further, I wrote the results, background and the literature review part in the report. As part of the team I helped in drawing inference and conclusion from the reported results.

References

[1] http://cs229.stanford.edu/proj2018/report/16.pdf

[2] https://www.tandfonline.com/doi/pdf/10.1080/09298215.2014.881888?needAccess=true

[3] Guo, A. Python API for Billboard Data. Github.com. Retrieved from: https://pypi.org/project/billboard.py/

[4] https://arxiv.org/abs/1710.10814

[5] https://pdfs.semanticscholar.org/8c03/0a736512456e9fd8d53763cbfcac0c014ab3.pdf

[6] https://ismir2014.ismir.net/LBD/LBD12.pdf

[7] https://opendatascience.com/a-machine-learning-deep-dive-into-my-spotify-data/

[8] Dorien Herremans, David Martens Kenneth Sörensen (2014) Dance Hit Song Prediction, Journal of New Music Research, 43:3, 291-302, DOI: 10.1080/09298215.2014.881888

[9] https://techxplore.com/news/2019-09-spotify-songs.html

[10] `http://www.maikaisogawa.com/wp-content/uploads/2019/07/CS_221_Predicting_the_Popularity_of_Top_100_Billboard_Songs.pdf`

[11] https://pdfs.semanticscholar.org/e6cc/edb50d2c2b01bca108cb090943e86fb58135.pdf

[12] https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28 (techniques for imbalanced classification)

[13] https://spotipy.readthedocs.io/en/2.11.2/

[14] http://millionsongdataset.com/