Open IIT

Data Analytics

Team 46

INDEX

1. INTRODUCTION……………………………………………………….......(3)
2. DATA DESCRIPTION………………………………………………....(4)
3. DATA ENGINEERING……………………………………………......(5)
4. DATA VISUALIZATION……………………………………………..(8)
    - FEATURE ENCODING……………………………………..(6)
5. EXPLORATORY DATA ANALYTICS………………………………....(7)
6. OBSERVATION…………………………………………………….....(8)
    - HEAT MAP……………………………………………………...(8)
    - SCATTER PLOT MATRICES…………………………..…..(9)
    - ALL SONGS………………………………………….…..……..(9)
    - ALL SONGS WITH HIGH POPULARITY…………...……(10)
    - RADAR CHART……………………………………………......(10)
7. APPROACH AND MODEL……………………………………………(13)
    - ML APPROACH………………………………………………...(13)
    - NEURAL NETWORK……………...…………………………..(13)
    - K MEANS……………………..……………………………..(13)
    - SVM CLASSIFIER……………………………………………..(13)
    - GAUSSIAN BIAS…………………………………………………(13)
    - DECISION TREE…………………………………………………(13)
    - RANDOM FOREST……………………………………….…...(14)
    - XG BOOST………………………………………………….......(14)
    - ADABOOST……………………………………………………….(14)
    - CAT BOOST…………………………………………………....(15)
    - GRADIENT BOOSTING……………………………………....(15)
    - GRID SEARCH………………………………………………...(15)
8. FINAL MODEL APPROACH…………………………………...…….(15)
9. REFERENCE…………………………………………………………..(17)

INTRODUCTION:

This report outlines the analysis performed on song popularity based on a number of characteristics of the song. The purpose of this analysis is to predict the popularity of the music tracks based on the features provided in the dataset and based on that predictions, bidding $10000 (in 10k) on 4000 music tracks and generate maximum revenue.

For the wrong prediction, bidding will be successful only if we bid on a less popular music track at the cost of a more popular music track. Vice versa is not possible.

Dataset given for this competition has a list of song characteristics consisting of different features like acousticness, danceability, energy, liveness, duration of the song, etc. We then use a number of machine learning algorithms (K-means, SVMs, Gaussian Naive Bayes, Neural networks, Random forest classifier, XG Boost etc) to output whether or not the song is popular.

The target variable, “popularity”, has 5 categories: ‘Very high’, ‘high’, ‘average’, ‘low’, ‘very low’. For each category, there is an initial bid price (for royalties to be paid) and expected revenue collections (in 10k $) as follows:

Popularity	**Bid Price**	**Expected Revenue**
Very high	5	10
High	4	8
average	3	6
low	2	4
Very low	1	2

DATA DESCRIPTION:

There are total of **16 features** present in the dataset and those are as follows:

- **Id-** Id column has a unique value for each song.
- **Acousticness-** A confidence measure between 0 and 1 of how acoustic a track is.
- **Danceability-** Describes how suitable a track is for dancing.
- **Energy-** A value representing a perceptual measure of intensity and activity.
- **Explicit-** An explicit track is one that has curse words or language or art that is sexual, violent or offensive in nature.
- **Instrumentalness**- Predicts whether a track contains no vocals.
- **Key-** The key the track is in. Integers map to pitches using standard Pitch Class notation.
- **Liveness-** Detects the presence of an audience in the recording
- **Loudness-** The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing the relative loudness of tracks
- **Mode-** Describes the modality (major or minor) of a track, the type of scale from which its melodic content is derived.
- **Release_date-** Date at which song is released.
- **Speechiness-** Detects the presence of spoken words in a track. The more exclusively speech-like the recording is, the closer the value is to 1.0.
- **Tempo-** The overall estimated tempo of a track in beats per minute (BPM).
- **Valence-** Describes the musical positiveness conveyed by a track.
- **Year-** Year in which song is/was released.
- **Duration-min-** The duration of a track in minutes.
- **Popularity-** Popularity of the song rating from very high to very low.

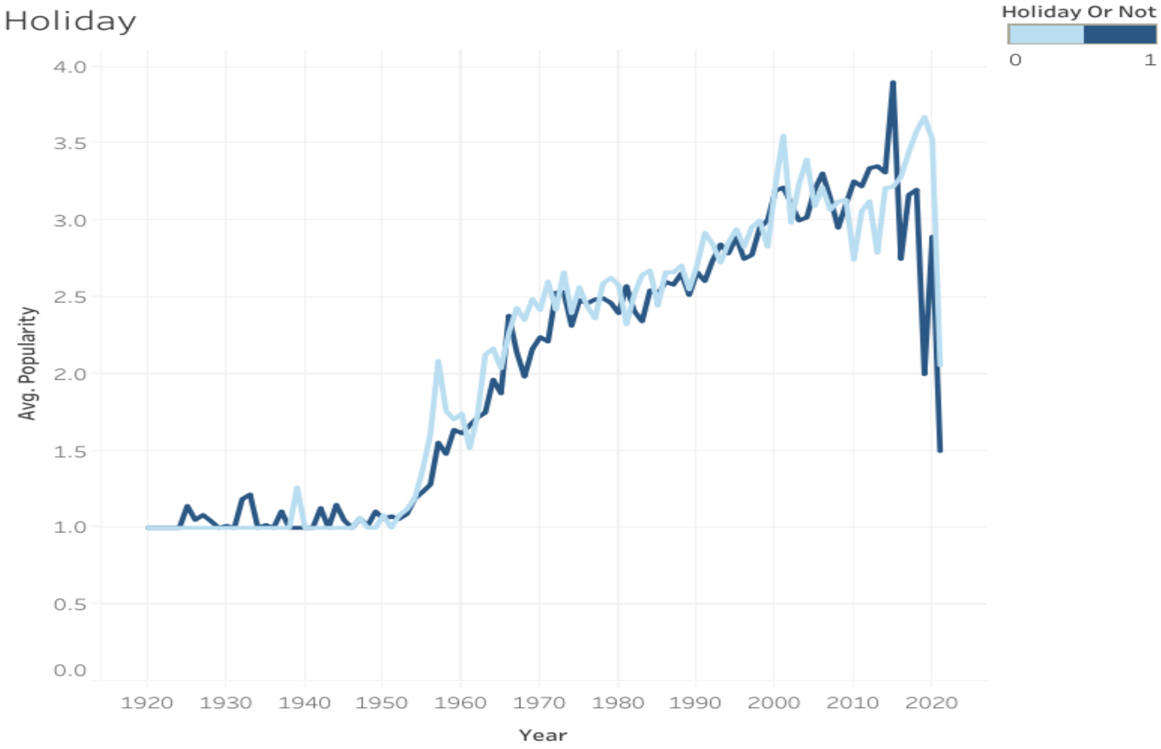
Popularity is our target variable and the rest are the independent variables.
The training data given have **12227** entries. Also, the dataset provided is void of any missing values.

DATA ENGINEERING:

The dataset provided was free from the null values so, there was no requirement of cleaning the dataset. In order to get better scoring, we designed a few new features using the existing dataset.

In order to obtain better revenue, we created a list of new features using the timestamp after proper research on what factors might affect the popularity of the music tracks. We extracted the date to determine whether it was a holiday or not, and a month to determine the season and a column of the year was already present in the dataset.

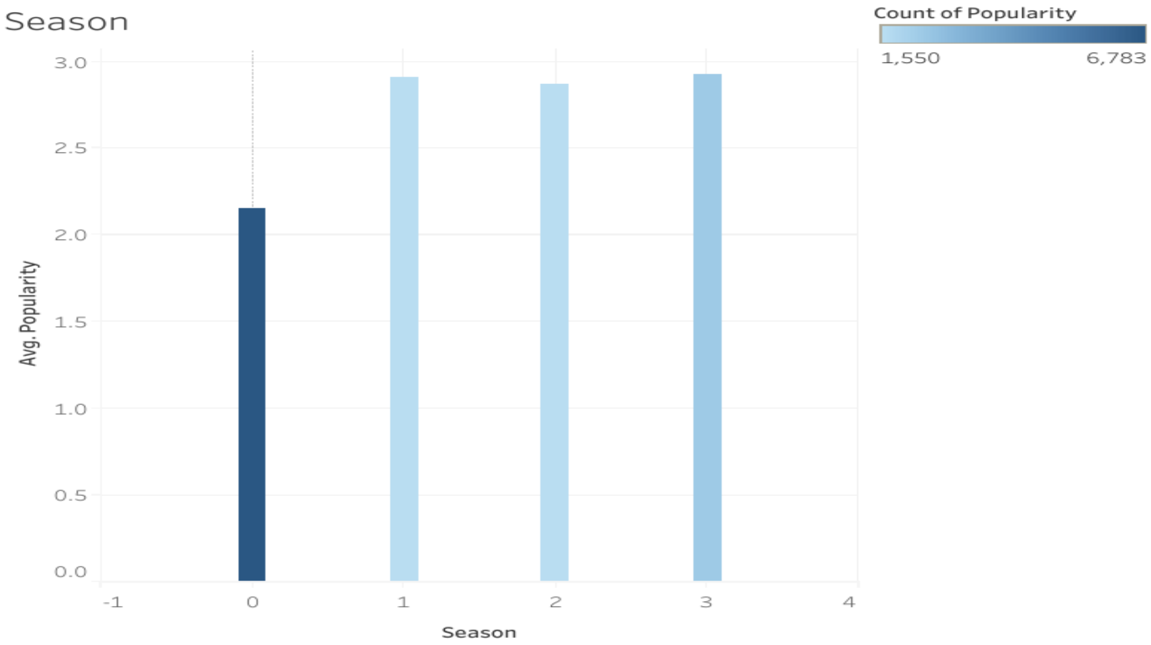
- Holiday- using python library "holidays" we were able to classify whether the particular day was a holiday or not.



    - As expected Average Popularity of a song is more on a holiday,the slight disparity at around 2000-2010 is due to less number of songs released on Holiday.

- Season- we divided the year into 4 seasons on the basis of month

  0 - Winter
  1 - Spring
  2 - Summer
  3 - Autumn



Majority of the songs are released in Winter i.e Season 0.

Feature Encoding:

There are some features in the dataset which required Label encoding to convert the labels into numeric form so as to convert it into the machine-readable form. Machine learning algorithms can then decide in a better way on how those labels must be operated. It is an important pre-processing step for the structured dataset in supervised learning.

Below are the features which required Label Encoding:

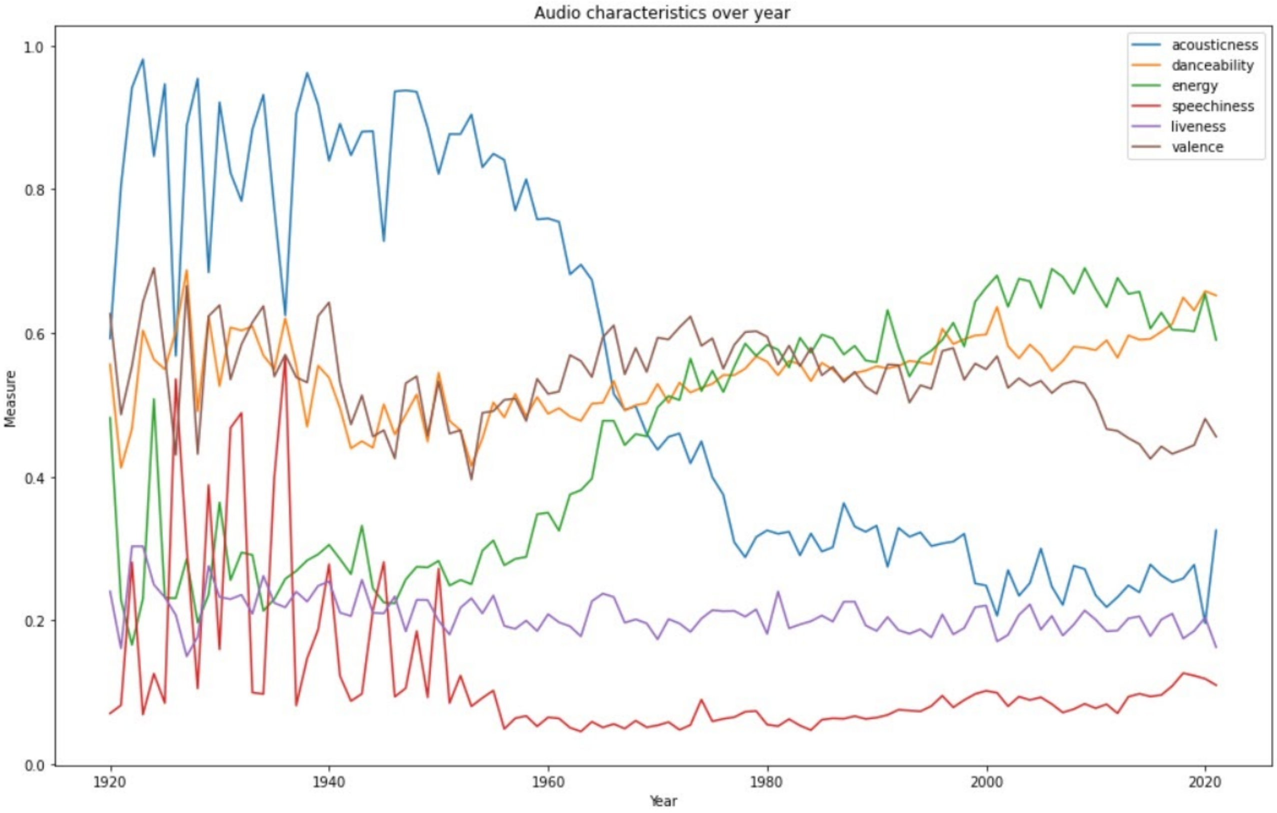
- Explicit
- Mode

One of the features given in the dataset was dropped as it was merely an index value while the relevant information was extracted from the other and those are:

- id
- Release_date

DATA VISUALIZATION:

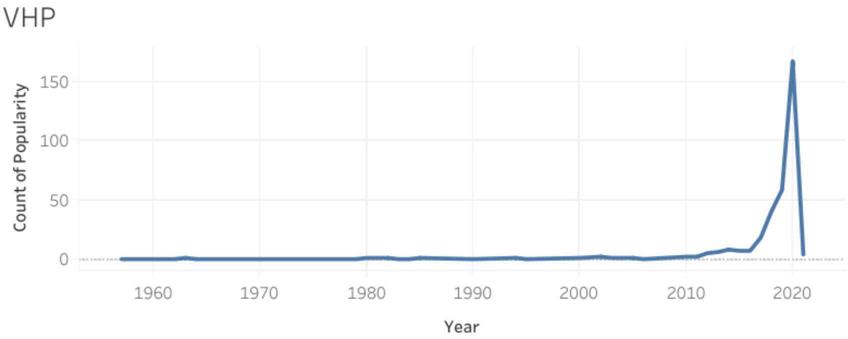
A Line graph was plotted to analyse audio characteristics over years



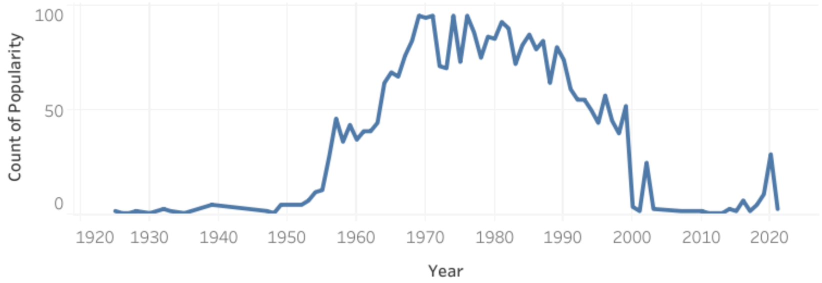
Observation:

- Acousticness has decreased whereas energy and danceability has increased on average over the years.

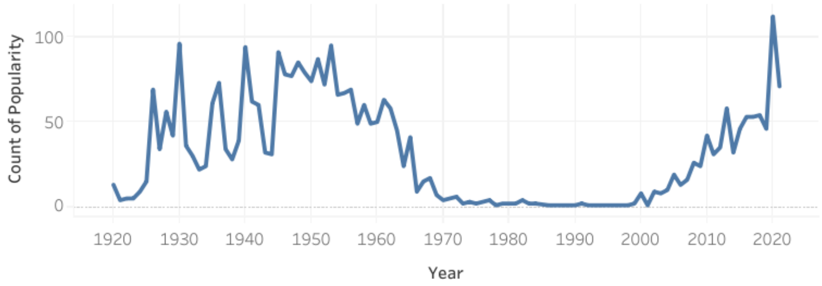
Below are the graphs showing the number of songs with different popularities vs year of release.



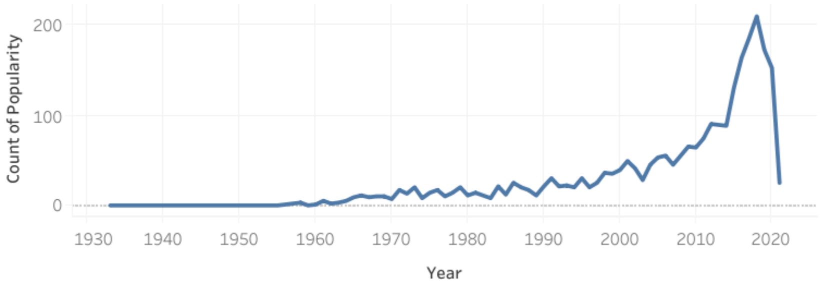
LP



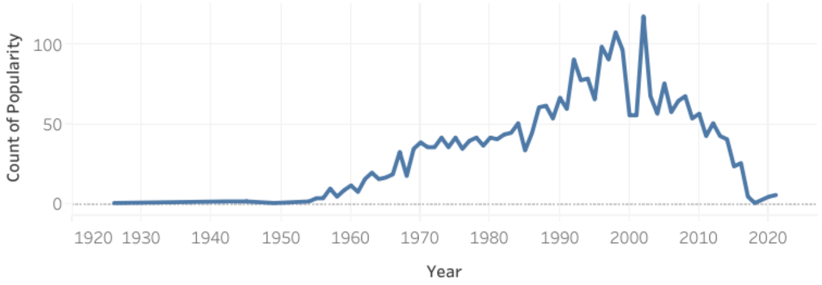
VLP



HP



AP



Observations:

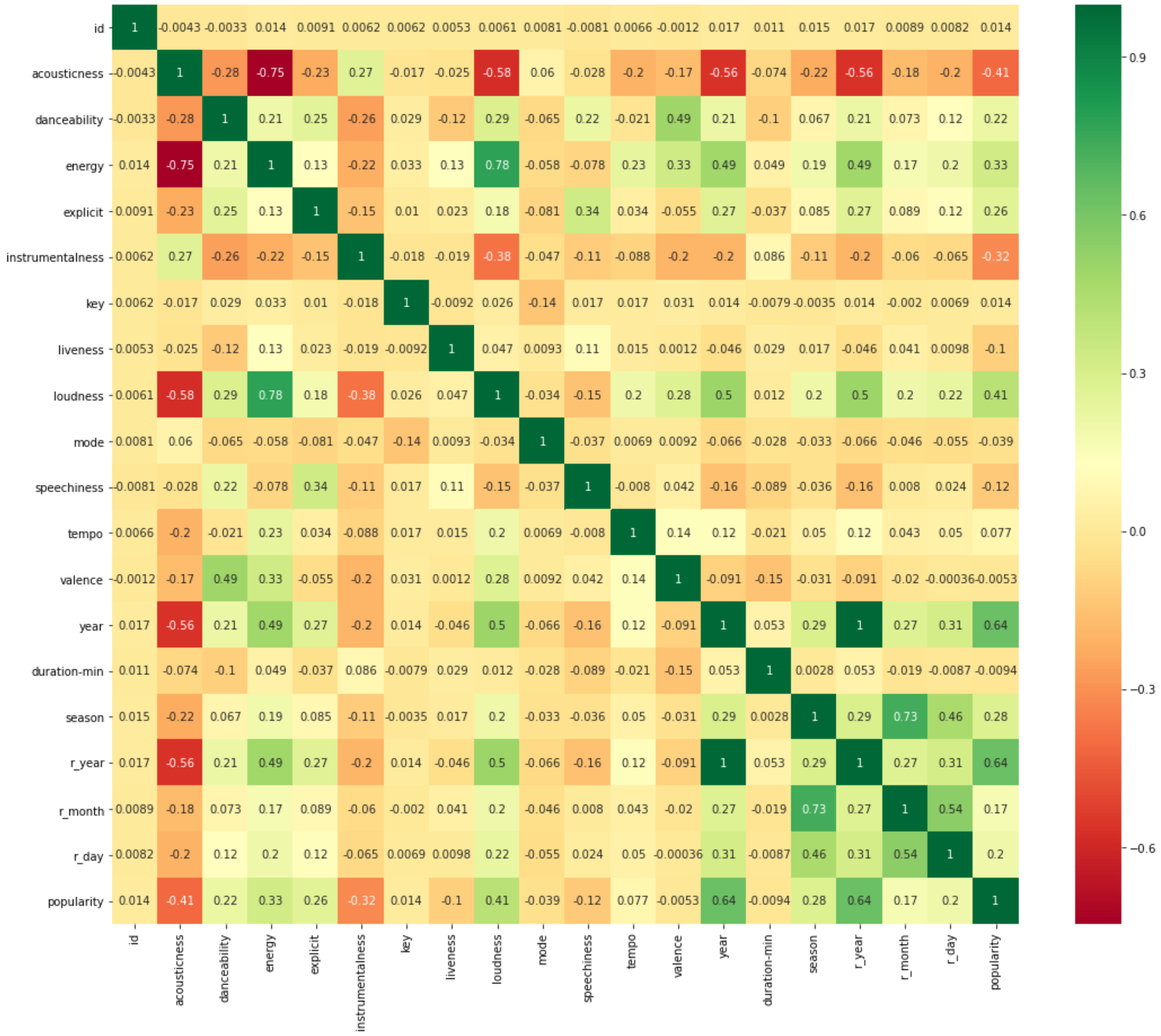
- Songs with high and very high probability have been released in recent years.

EXPLORATORY DATA ANALYSIS:

With a focus on summarizing and visualizing the important characteristics of a data set, exploratory data analysis assists in understanding the data's underlying structure and variables, developing intuition about the data set and deciding how it can be investigated with more formal statistical methods. After a detailed exploratory analysis, we gathered some significant results.

Heat Map :

The correlation matrix was plotted in order to ensure the new features created don't overlap with each other or give erroneous results.



Observation:

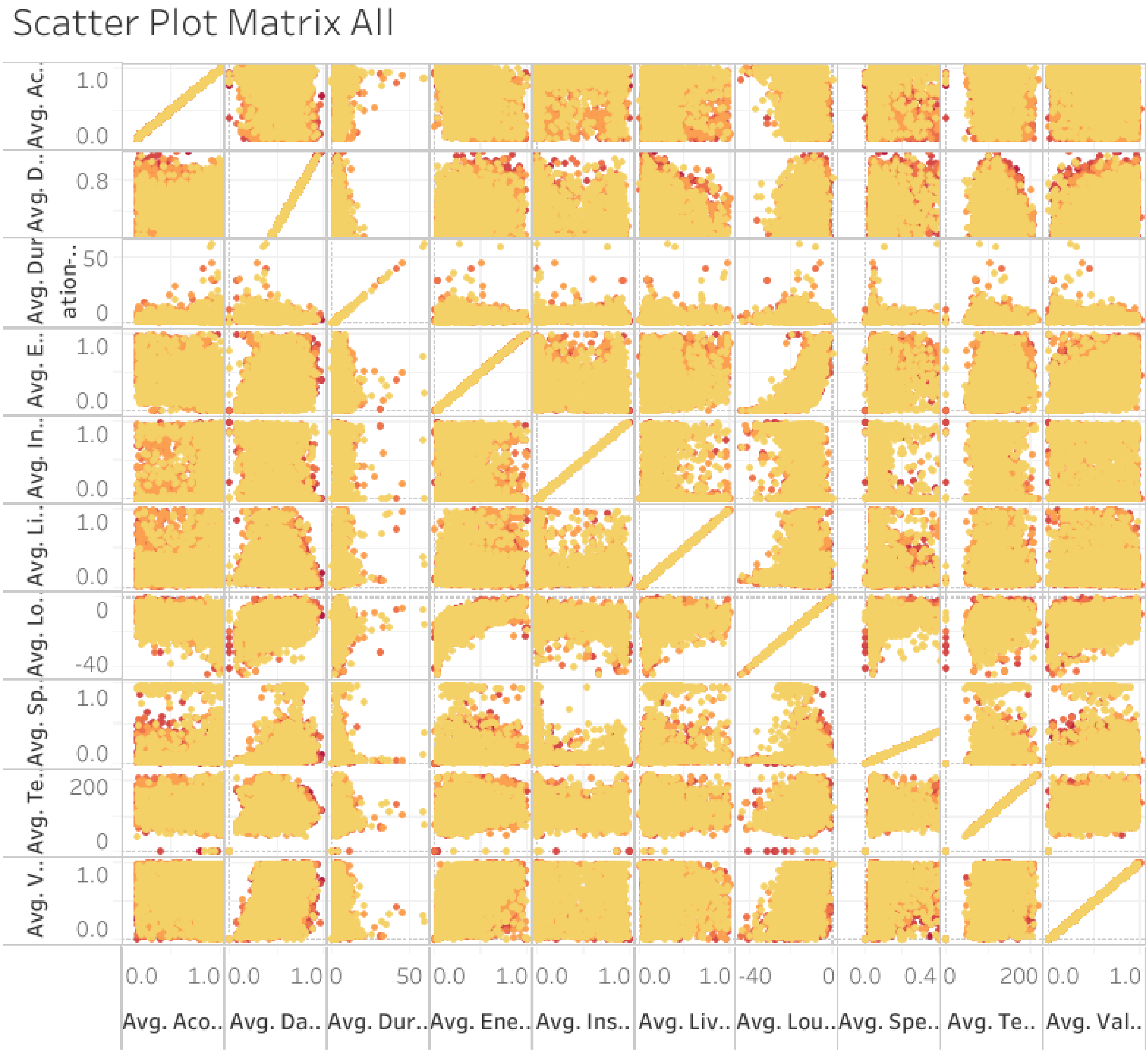
- By referring to the heat map we r_year due to overlapping with year.
- We also identified that year is highly correlated with popularity as seen in data visualization.

Scatter Plot Matrices:

Scatter Plot Matrices provide a quick visual way to check possible correlations between various variables and to obtain useful information which can be used in data analysis.

Below are the scatter plot matrices considering songs with different popularity.

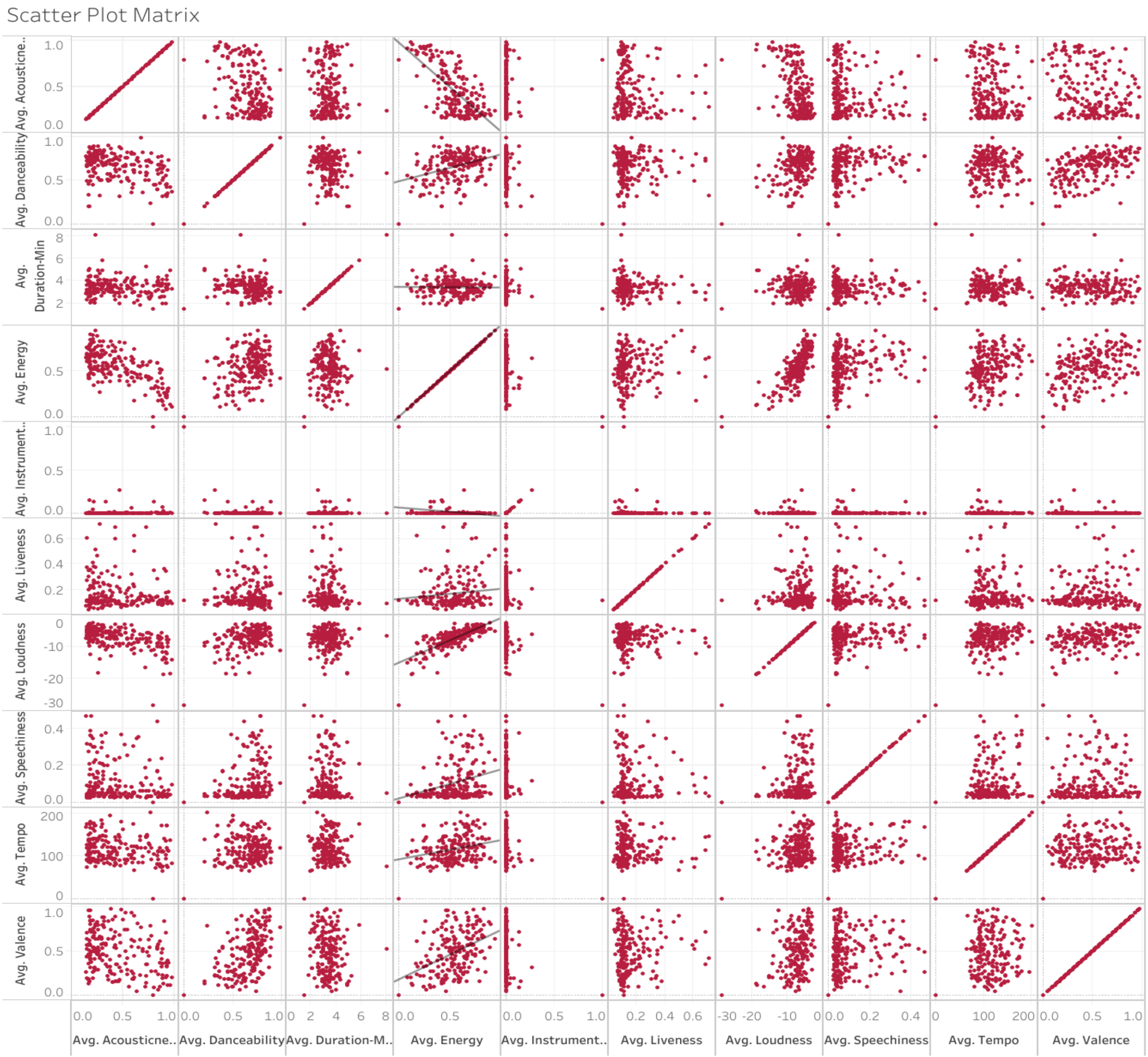
All Songs



Observations:

- Duration is almost constant for all songs and doesn’t have a strong relation with any other parameter
- There seems to be a relation between Loudness and energy of a song,which is more clearly observed in other plots

Songs with Very High Popularity



Observations:

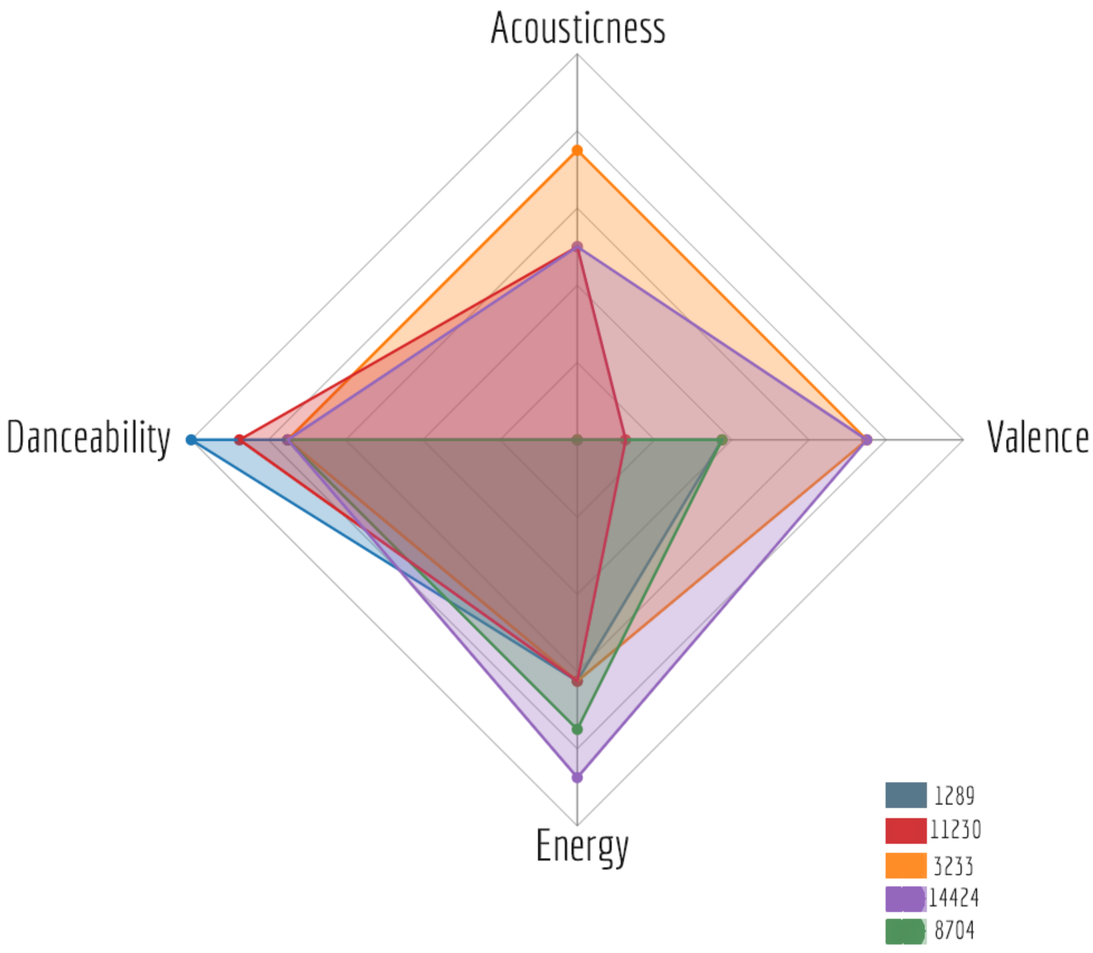
- Songs with Very High popularity tend to have less instrumentalness and speechiness

Similarly Scatter Plot Matrices were used to observe trends with songs of different popularity.

Radar Charts

Radar charts provide another useful visual way to observe trends with various variables.

Radar Charts for songs with Very High popularity were observed in groups of 5. One such group is shown below, with legend indicating song id.



Observations:

- Songs with Very High popularity tend to have high danceability and energy

APPROACH AND MODELS-

ML Approach:

Machine learning (ML) is concerned with algorithms and techniques that allow computers to learn. The ML approach covers main domains, such as data mining, difficult to program applications, and software applications. It is a collection of a variety of algorithms that can provide multivariate, nonlinear, nonparametric regression or classification. Machine learning algorithms use computational methods to "learn" information directly from data without relying on a predetermined equation as a model

Neural Network:

The structure of the human brain inspires a Neural Network. It is essentially a Machine Learning model (more precisely, Deep Learning) that is used in unsupervised learning.
A Neural Network consists of an assortment of algorithms used in Machine Learning for data modelling using graphs of neurons.
For the model implementation (training), we used 2 hidden layers of sizes (6x1) each, the input layer (No. of train cases x 1) and the output as the popularity (5x1).
Also, "categorical_crossentropy" was taken to be the loss function, and "relu" activation for the 2 hidden layers and "sigmoid" for the output layer.

K means:

The k-means algorithm is an unsupervised clustering algorithm. It takes a bunch of unlabeled points and tries to group them into "k" number of clusters. It is unsupervised because the points have no external classification.
The "k" in k-means denotes the number of clusters you want to have in the end. If k = 5, you will have 5 clusters on the data set.

SVM classifier:

Support Vector Machine" (SVM) is a supervised machine learning algorithm that can be used for both classification or regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is the number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well

Gaussian Bias:

Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. The technique is easiest to understand when described using binary or categorical input values.

It is called naive Bayes or idiot Bayes because the calculation of the probabilities for each hypothesis is simplified to make their calculation tractable. Rather than attempting to calculate the values of each attribute value P(d1, d2, d3|h), they are assumed to be conditionally independent given the target value and calculated as P(d1|h) * P(d2|H) and so on.

This is a very strong assumption that is most unlikely in real data, i.e. that the attributes do not interact. Nevertheless, the approach performs surprisingly well on data where this assumption does not hold.

Decision Tree:

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. DecisionTreeClassifier is a class capable of performing multi-class classification on a dataset. As with other classifiers, DecisionTreeClassifier takes as input two arrays: an array X, sparse or dense, of shape (n_samples, n_features) holding the training samples, and an array Y of integer values, shape (n_samples,), holding the class labels for the training sample

Random Forest:

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

Parameters used: "max depth” = 10, ”min_samples_split” = 8,”n estimator" = 500

XG Boost:

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. Using the “hyperopt” library we tuned the parameters of the XGboost model.

Parameters used: “learning rate” = 0.04, “n_estimators” = 450, “gamma” = 0, “max depth” = 5

Adaboost:

AdaBoost algorithm, short for Adaptive Boosting, is a Boosting technique that is used as an Ensemble Method in Machine Learning. It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights to incorrectly classified instances. Boosting is used to reduce bias as well as the variance for supervised learning. It works on the principle where learners are grown sequentially. Except for the first, each subsequent learner is grown from previously grown learners. In simple words, weak learners are converted into strong ones. Adaboost algorithm also works on the same principle as boosting, but there is a slight difference in working.

Parameters used: "learning rate" = 0.4, "n estimator" = 900

Catboost:

CatBoost is a recently open-sourced machine learning algorithm from Yandex. It can easily integrate with deep learning frameworks like Google's TensorFlow and Apple's Core ML. It can work with diverse data types to help solve a wide range of problems that businesses face today. To top it up, it provides best-in-class accuracy.

"CatBoost" name comes from two words "Category" and "Boosting".

As discussed, the library works well with multiple Categories of data, such as audio, text, image including historical data.

Gradient Boosting:

"Boost" comes from a gradient boosting machine learning algorithm as this library is based on a gradient boosting library. Gradient boosting is a powerful machine learning algorithm that is widely applied to multiple types of business challenges like fraud detection, recommendation items, forecasting and it performs well also. It can also return very good result with relatively less data, unlike DL models that need to learn from a massive amount of data.

Grid search:

Grid-searching is the process of scanning the data to configure optimal parameters for a given model. Depending on the type of model utilized, certain parameters are necessary. Grid-searching does NOT only apply to one model type. Grid-searching can be applied across machine learning to calculate the best parameters to use for any given model. It is important to note that Grid-searching can be extremely computationally expensive and may take your machine quite a long time to run. Grid-Search will build a model on each parameter combination possible. It iterates through every parameter combination and stores a model for each combination.

FINAL MODEL APPROACH-

Upon implementing the machine learning models namely (Random Forest, XG Boost, AdaBoost) we got nearly the same revenue earned but on analysing the confusion matrix we found out that most of the misclassification was where there was **under-classification**. On top of that bidding will be successful only if we bid on a less popular music track at the cost of a more popular music track, hence it would be desirable to classify the song one class above if not correct. Also, we were not able to maximize our bidding which had a ceiling of 2.5 times the number of datapoints. Thus we chose three models having nearly the same revenue generated (Random Forest, XGBoost, AdaBoost) and took the **ceiling value of the average of the three predictions**. Thereby even if one of the models predicted a certain datapoint to have a class one higher than the other two then we would bid assuming it to be such.

For example if the actual category of song was “high”, if two of the models predicted the song to be of category “average” and the other “high”. Then had we applied a single model we would have got a revenue of 2 * 4 = 8, on average it would be **$8/3**. But on applying a ceiling function on the predicted output we would have classified it correctly as “high” hence a revenue of **$8**.
Taking another example if the actual category was “high”, if two of the models predicted the song to be of category “high” and the other “average”. Then if had we applied a single model we would have got a revenue of 2 * (2 * 4) = 16, on average it would be **$16/3**. But on applying a ceiling function on the predicted output we would have classified it correctly as “high” hence a revenue of **$8**.

On the test data, our final model predicted:

Popularity	Count
Very Low	960
Low	1135
Average	1035
High	719
Very High	151

Assuming if we are able to bid correctly on every song, the total bid comes out to be **$9966** which is less than the maximum allowable limit of **$10000**.

REFERENCES-

https://kth.diva-portal.org/smash/get/diva2:1214146/FULLTEXT01.pdf

https://machinelearningmastery.com/naive-bayes-for-machine-learning/

https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/

https://towardsdatascience.com/understanding-gradient-boosting-machines-9be756fe76ab

https://towardsdatascience.com/grid-search-for-model-tuning-3319b259367e

https://towardsdatascience.com/an-intuitive-explanation-of-random-forest-and-extra-trees-classifiers-8507ac21d54b

https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

https://www.analyticsvidhya.com/blog/2017/08/catboost-automated-categorical-data/