Spotify Hit Predictor

A Thesis submitted to

Great Lakes Institute of Management

In partial fulfilment of requirements for

the award of

Post Graduate Programme in Data Science and Engineering

By

Indu Chahal,

Tushar Maheshwari,

Uday Kumar,

Venkatesh Gundu,

Zeba Anjum

Under guidance of

Arpit Sharma

Table of Contents

PROBLEM STATEMENT

The Billboard Hot 100 Chart remains one of the definitive ways to measure the success of a popular song. We investigated using machine learning techniques to predict whether or not a song will become a hit, based on its audio features.

We are using a dataset consisting of approximately 40,000 hit and non-hit songs along with their audio features extracted from the Spotify Web API.

The input to each algorithm is a series of audio features of a track. We use the algorithm to output a binary prediction of whether or not the song will feature on the Billboard Hot 100.

For this project we hope to gain the following insights:

1. **Predict if the songs will be a hit or not.**
2. **What songs with which artist are getting more hits?**
3. **What type of beats are in the hit list?**
4. **What kind of songs are popular with respect to lyrics?**

PROJECT OUTCOME

The outcome of our project is to build a robust machine learning algorithm that helps the business client understand what are the factors in a song, that helps a song to be a hit.

INDUSTRY REVIEW

The music industry can be best described as an imperfect art. The expression of sound through instruments and voices is at its core an experimental process, with no correct answers. The lack of an absolute 'correct' is certainly frustrating to all stakeholders in the industry: artists, fans, managers, record label executives.

What if there was a way to determine if a song would be a hit before the song was released to the world? The money would be saved for the stakeholders in the production process, artists would not release songs that resulted in commercial failure.

This research is therefore relevant to musicians and music labels. Not only will it help determine how best to produce songs to maximise their potential for becoming a hit, but it could also help decide which songs could give the greatest return for investment on advertising and publicity.

Furthermore, it would help artists and music labels determine which songs are unlikely to become Billboard Hot 100 hits.

DATA SET

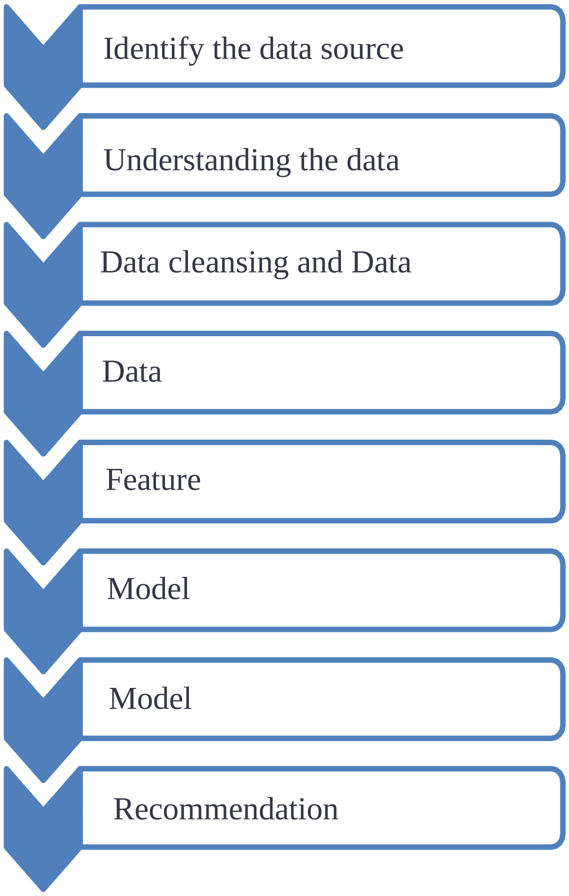
- A dataset is a collection of data, and it can be structured or unstructured.
- A structured data is represented in a tabular format, where every column of the table represents a particular variable, and each row corresponds to a given record of the dataset.
- Unsupervised/unstructured data is not represented in a tabular form, data that we fetch from Kaggle.

DATA DESCRIPTION

This is a dataset consisting of features for tracks fetched using Spotify's Web API. The dataset has 41106 rows with 20 features. Refer to the below-detailed structure of the dataset.

Variable Name	Variable Description
1. Track	The name of the track.
2. Artist	The name of the artist.
3. decade	The decade the song belongs.
4. Uri	The resource identifier for the track.
5. Danceability	Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity.
6. Energy	It represents a perceptual measure of intensity and activity.
7. key	Integers map to pitches using standard Pitch Class notation.
8. loudness	The overall loudness of a track in decibels (dB).
9. mode	Mode indicates the modality (major or minor) of a track.
10. speechiness	Speechiness detects the presence of spoken words in a track.
11. acousticness	A confidence measure from 0-1 of whether the track is acoustic.
12. instrumentalness	Predicts whether a track contains no vocals.
13. liveness	Detects the presence of an audience in the recording
14. valence	A measure from 0-1 describing the musical positiveness conveyed by a track.
15. tempo	The overall estimated tempo of a track in beats per minute (BPM).
16. duration_ms	The duration of the track in milliseconds.
17. time_signature	An estimated overall time signature of a track.
18. chorus_hit	This the author's best estimate of when the chorus would start for the track.
19. sections	The number of sections the particular track has.
20. target	The target variable for the track - 0 or 1.

PROJECT METHODOLOGY



DATA PRE-PROCESSING

Data Preparation:

Data pre-processing is a crucial step that helps enhance the quality of data to promote the extraction of meaningful insights from the data. Data pre-processing in Machine Learning refers to the technique of preparing (cleaning and organising) the raw data to make it suitable for building and training Machine Learning models.

Data Preparation is the process of collecting, cleaning, and consolidating data into one file or data table, primarily for use in the analysis.

Acquire the dataset: https://www.kaggle.com/datasets/theoverman/the-spotify-hit-predictor-dataset

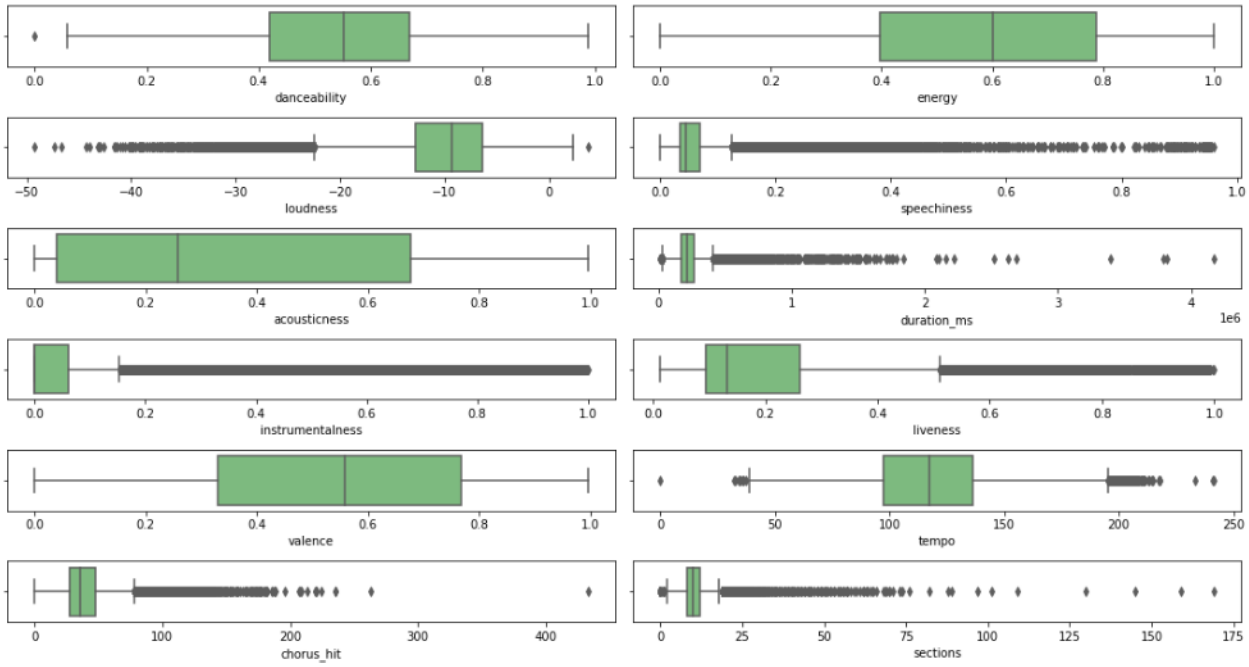
Missing/Null Values:

Impute or drop features with missing values based on the percentage of missing values and relevance for model building.

In the dataset there are no null values.

OUTLIERS:

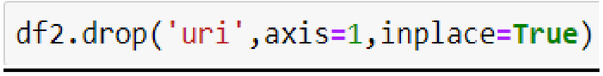
Outliers are the extreme that deviates from other observations on data, they may indicate variability in measurement, experimental errors or a novelty.



We have a significant number of outliers present in most columns. We may not treat them for the base model, to check the significance of each independent feature as based on their significance we could tune the model in order to build a better model.

REDUNDANT COLUMNS

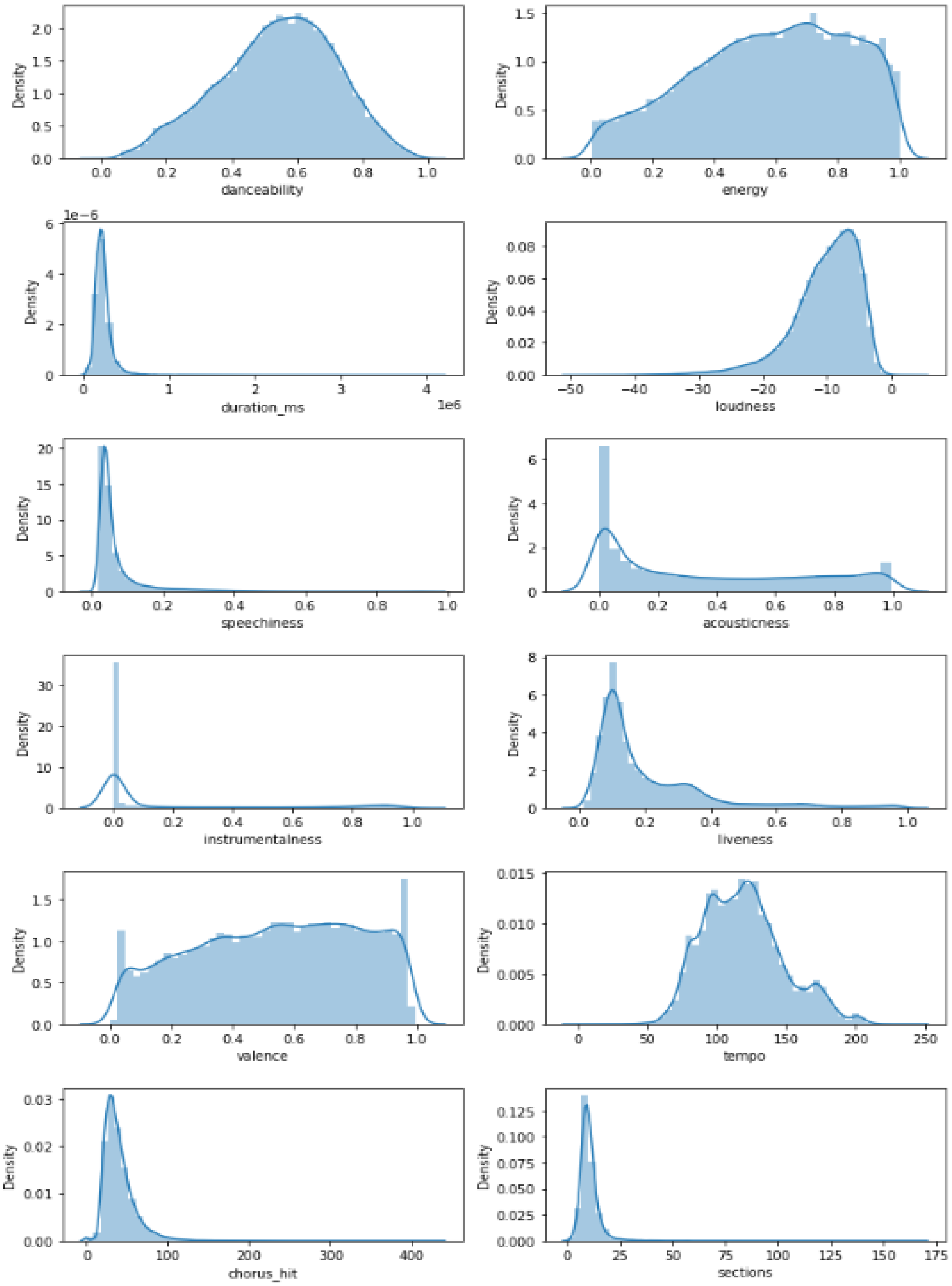
We have one redundant column in the dataset which is 'uri' and hence we have decided to drop it.



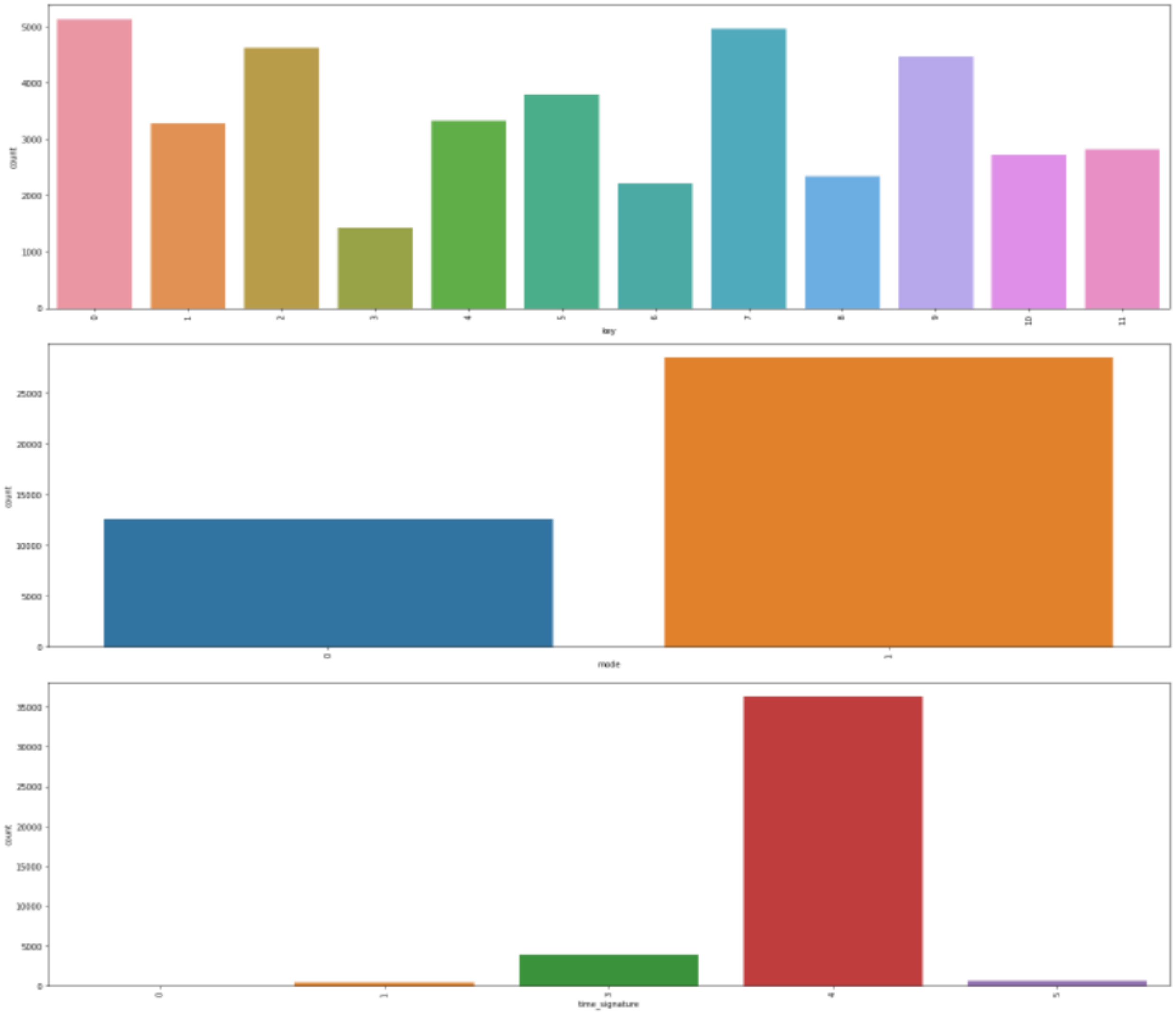
```
df2.drop('uri',axis=1,inplace=True)
```

EXPLORATORY DATA ANALYSIS & BUSINESS INSIGHTS

Univariate Analysis:

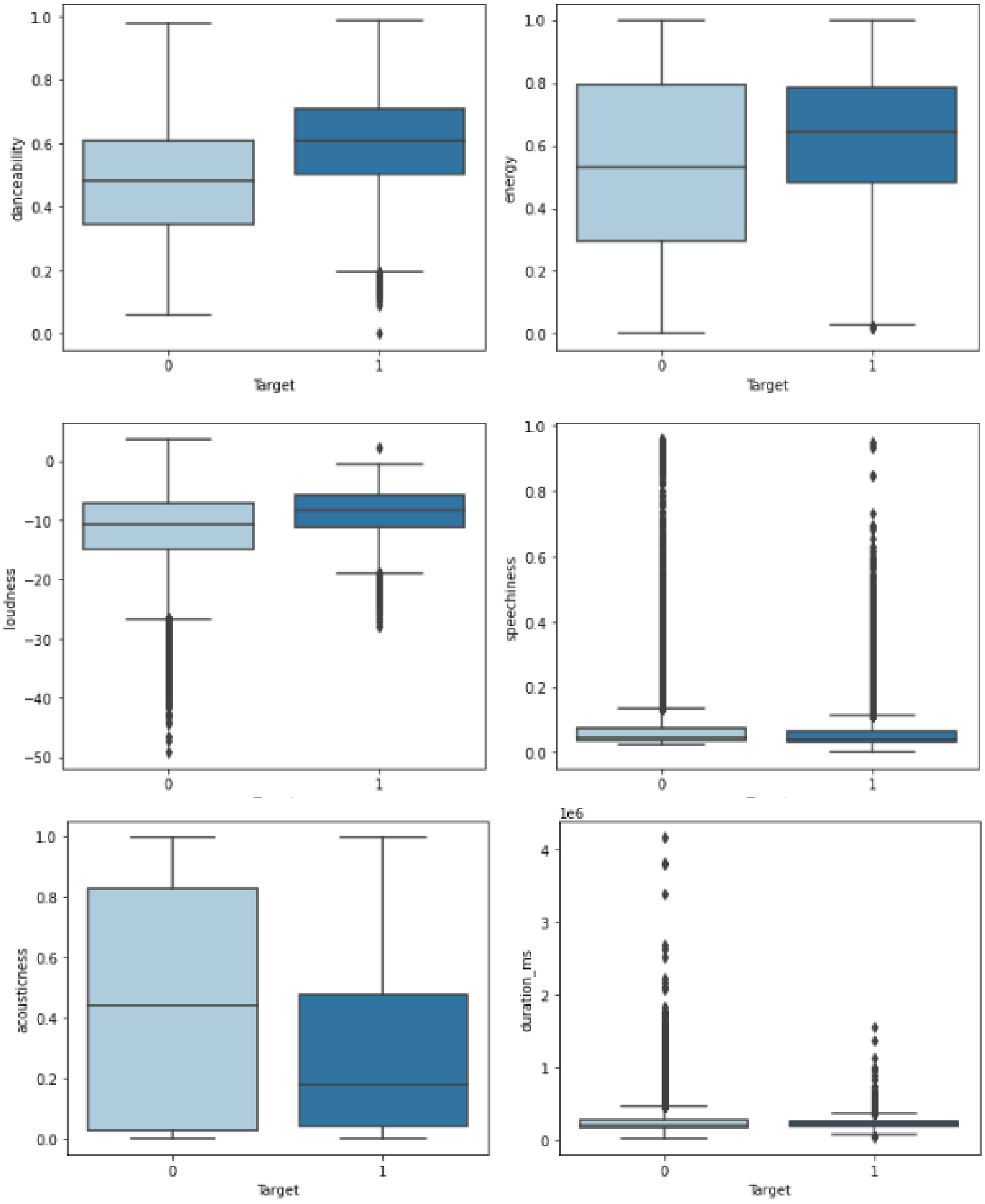


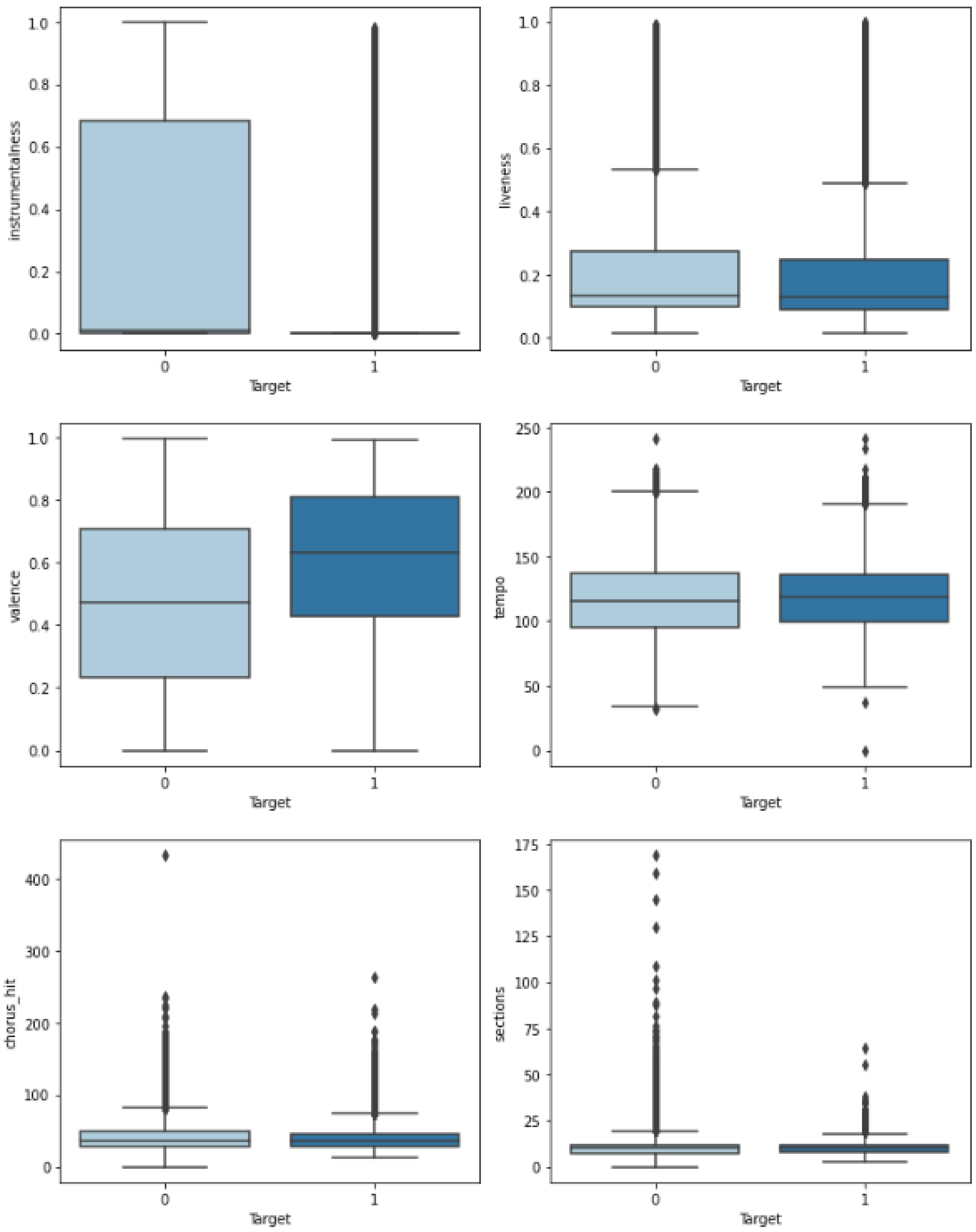
As per the above analysis there are independent variables (danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, tempo, chorus hit and sections) in the data set, most of which are slightly skewed. These features shall be transformed as we proceed towards model building.



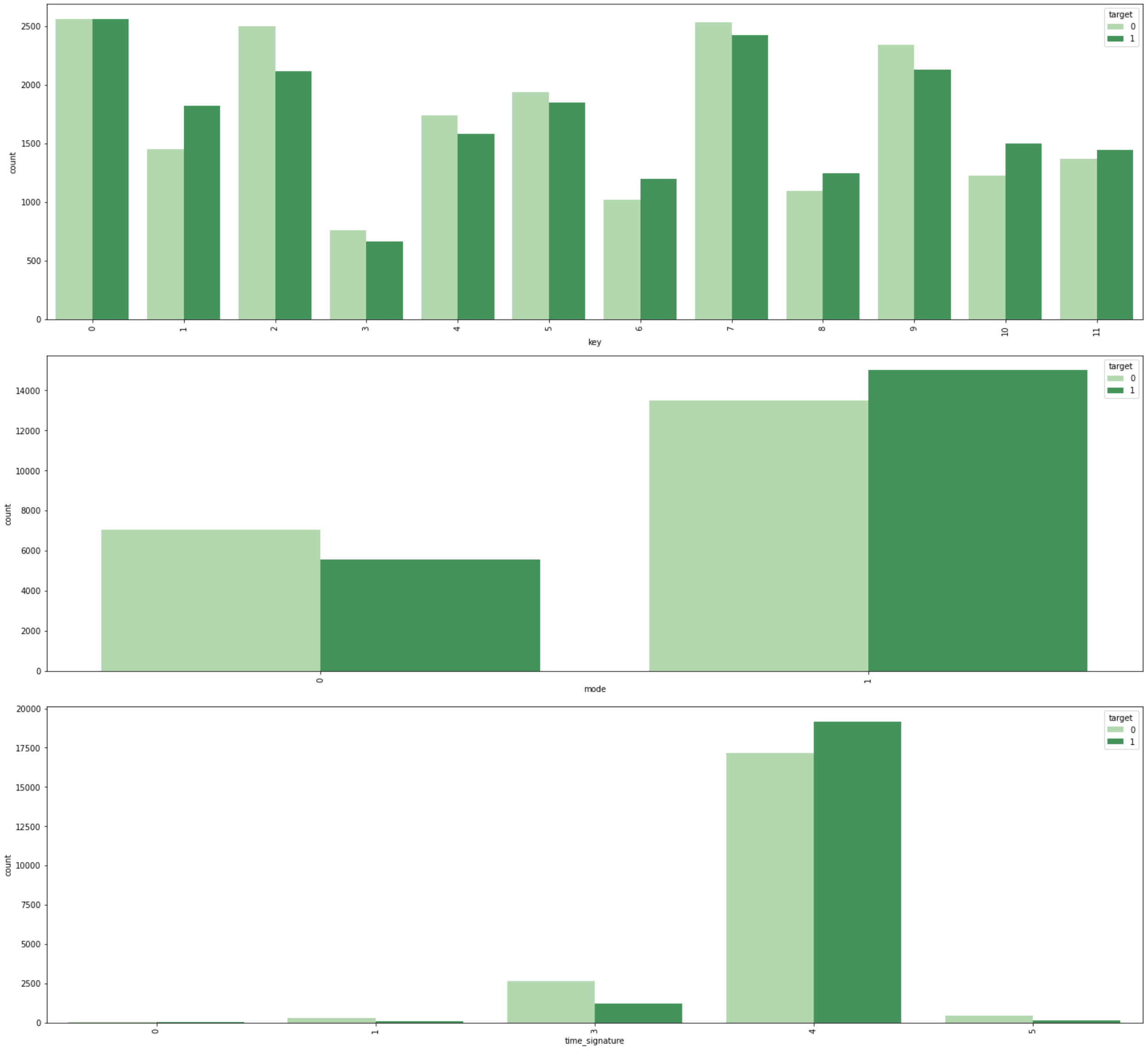
The above graphs are the count plots for each independent categorical variable (key, mode and time signature) that essentially represents the number of songs that fall into each subclass of the independent categorical variables.

Bi-variate analysis with target feature:





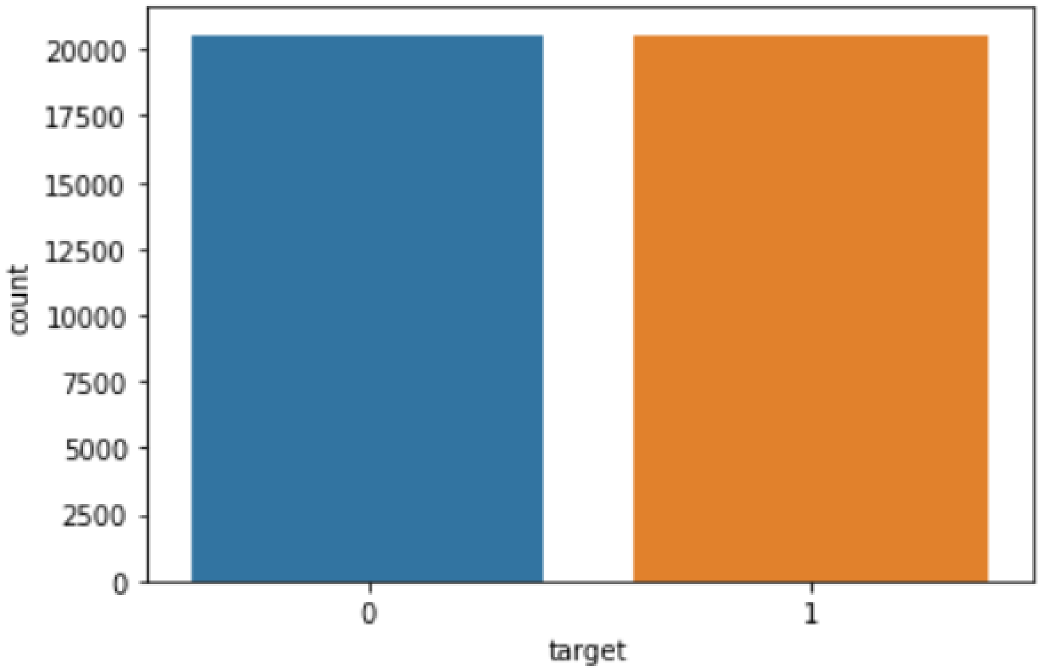
The above is the bivariate analysis of independent numeric columns which are danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, tempo, chorus_hit and sections with the target variable. Here, it can be inferred that the variables have a significance in determining if a song could be a hit or not.



The above is the bivariate analysis of independent categorical columns which are key, mode and time signature with the target variable. It can be inferred that in most sub categories of the categorical variables, the distribution of songs is not exactly equal with respect to the target variable.

Univariate analysis of target feature:

By analysing the target feature, the data is perfectly balanced.



STATISTICAL TESTING:

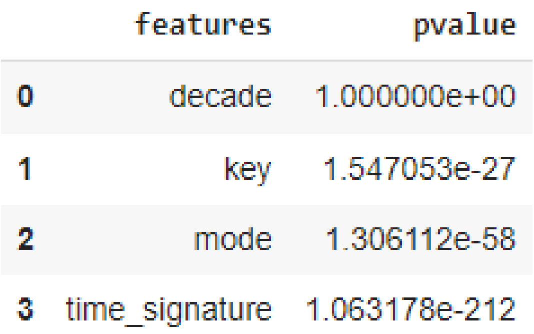
Chi-Square test of Independence:

- The Chi-Square test of independence is used to determine if there is a significant relationship between two nominal (categorical) variables.
- The hypothesis to test the independence of attributes

H_0: The attributes are independent

against

H_a: The attributes are dependent

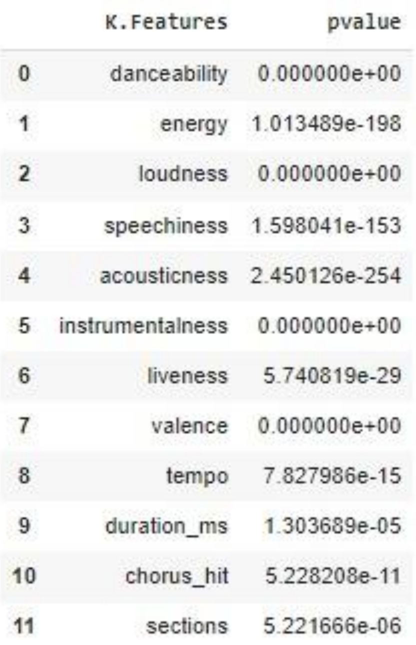


	features	pvalue
0	decade	1.000000e+00
1	key	1.547053e-27
2	mode	1.306112e-58
3	time_signature	1.063178e-212

Chi-square independence test on decade, key, mode, time_signature and target category features:

- It is observed that Decade has the pvalue as 1.0 which is more than 0.05 significance level, hence we accept the null hypothesis. Hence there is no significant impact of decade on target variable.
- It is observed that key, mode, time signature has the pvalue less than 0.05 significance level, hence we fail to accept the null hypothesis. Hence there is significant impact of decade on target variable.

KRUSKAL-WALLIS:



	K.Features	pvalue
0	danceability	0.000000e+00
1	energy	1.013489e-198
2	loudness	0.000000e+00
3	speechiness	1.598041e-153
4	acousticness	2.450126e-254
5	instrumentalness	0.000000e+00
6	liveness	5.740819e-29
7	valence	0.000000e+00
8	tempo	7.827986e-15
9	duration_ms	1.303689e-05
10	chorus_hit	5.228208e-11
11	sections	5.221666e-06

KRUSKAL-WALLIS TEST ON TARGET AND OTHER NUMERICAL FEATURES:

Inferences

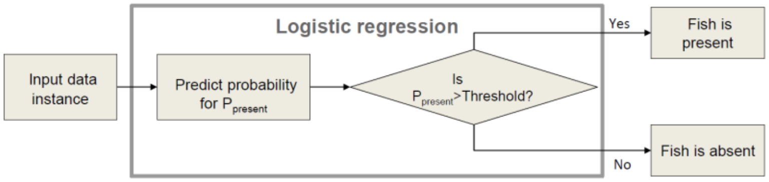
- It is observed that the pvalue for all independent numeric variables is less than 0.05 significance level, hence we fail to accept the null hypothesis.
- We are doing this test because Shapiro and levene tests are failing and Anova test cannot be performed. Since the data is skewed so we do Kruskal-Walis test
- This is showing the features where pval is less than alpha where the confidence interval is 95%. Those are the feature that have a relationship with target variable

MODEL BUILDING:

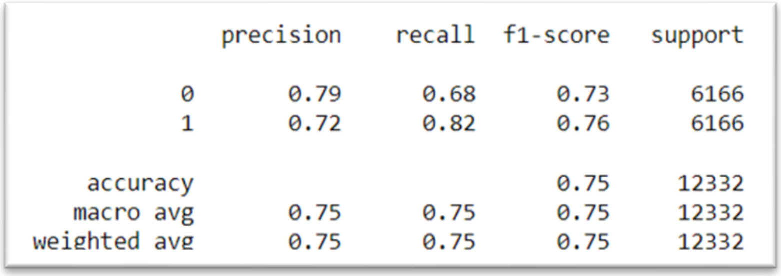
Logistic regression:

- Logistic Regression is a binary classification algorithm. It predicts the probability of occurrence of a label class.

- Consider that logistic regression is used to identify whether the product falls under the advantage category or not.

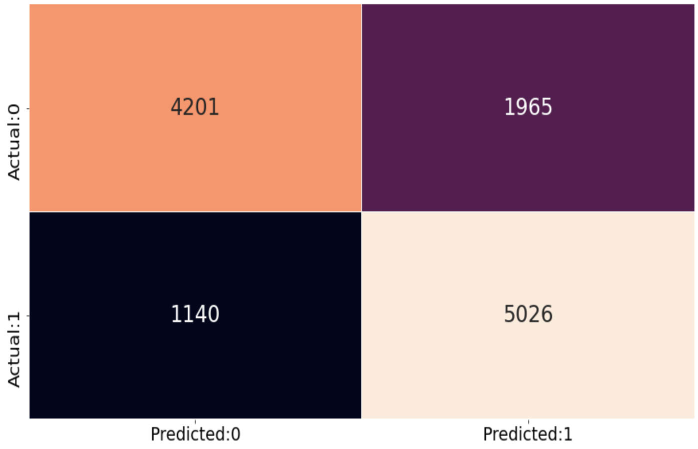


Classification report:

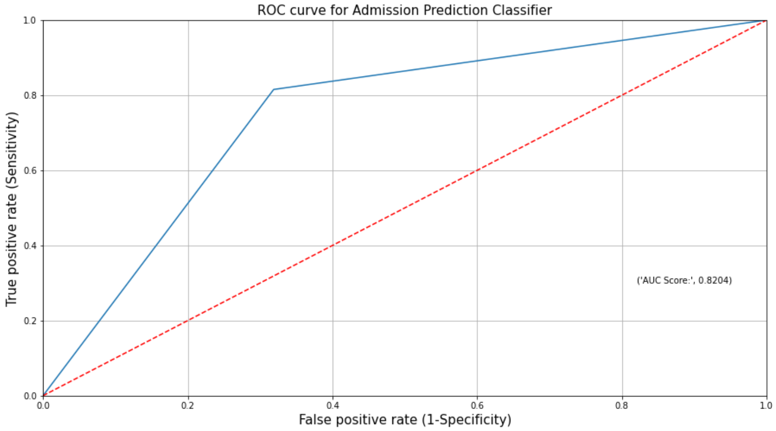


```
              precision    recall  f1-score   support

           0       0.79      0.68      0.73      6166
           1       0.72      0.82      0.76      6166

    accuracy                           0.75     12332
   macro avg       0.75      0.75      0.75     12332
weighted avg       0.75      0.75      0.75     12332
```

Confusion Matrix:

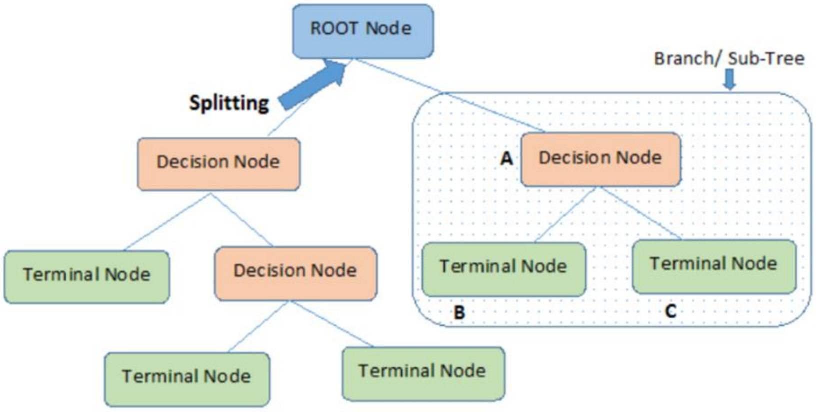


ROC Curve:

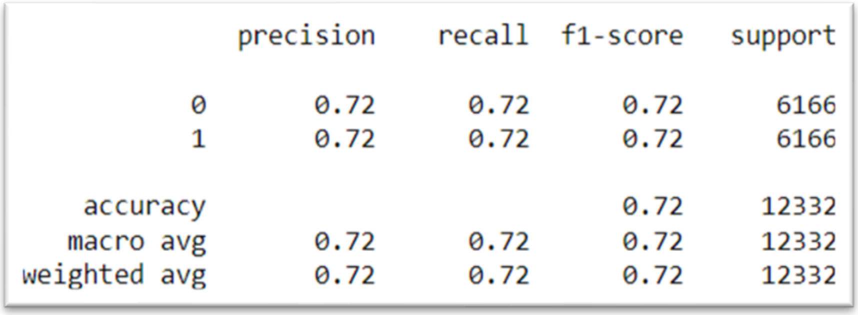


Decision Tree Algorithm:

- Decision trees can be used for classification as well as regression problems.
- The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits.
- It starts with a root node and ends with a decision made by leaves

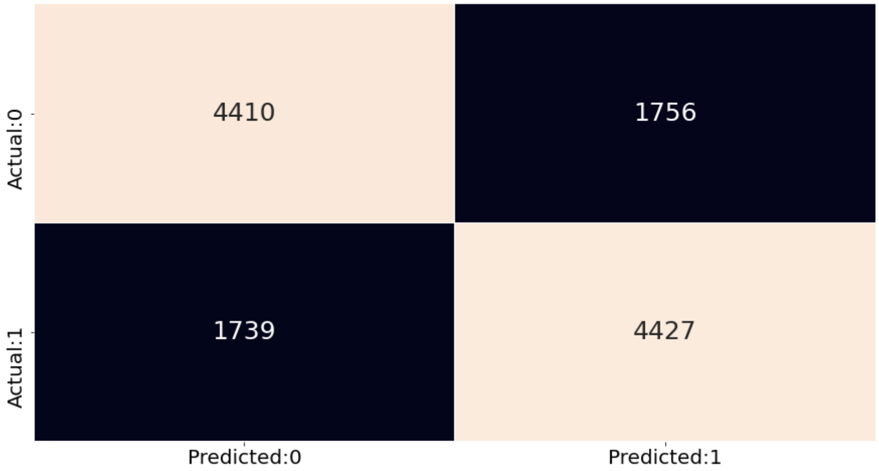


Classification Report:



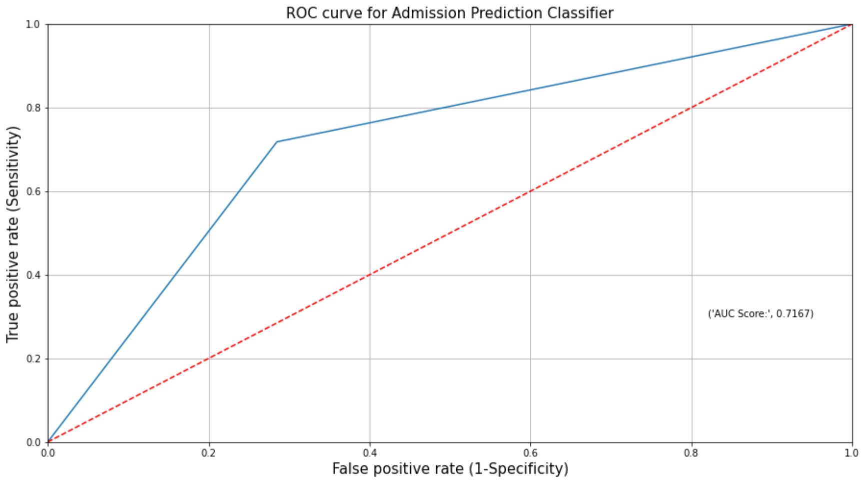
```
              precision    recall  f1-score   support

           0       0.72      0.72      0.72      6166
           1       0.72      0.72      0.72      6166

    accuracy                           0.72     12332
   macro avg       0.72      0.72      0.72     12332
weighted avg       0.72      0.72      0.72     12332
```

Confusion Matrix



	Predicted:0	Predicted:1
Actual:0	4410	1756
Actual:1	1739	4427

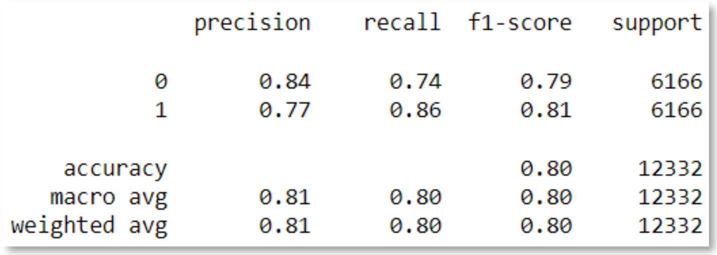
ROC Curve:



Random Forest Algorithm:

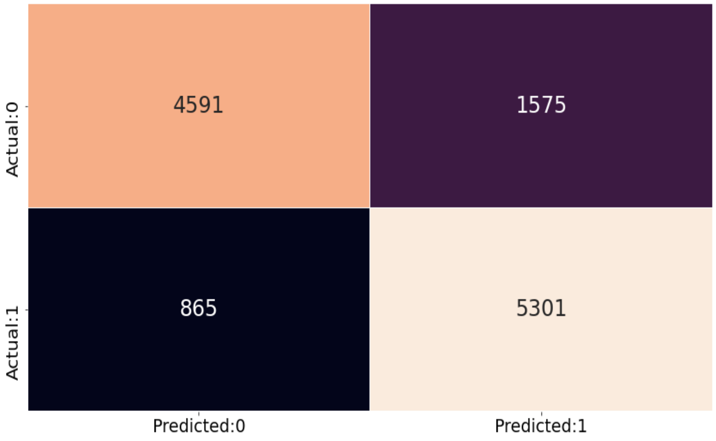
- Random Forest consists of several independent decision trees that operate as an ensemble.
- It is an ensemble learning algorithm based on bagging.

Classification Report:

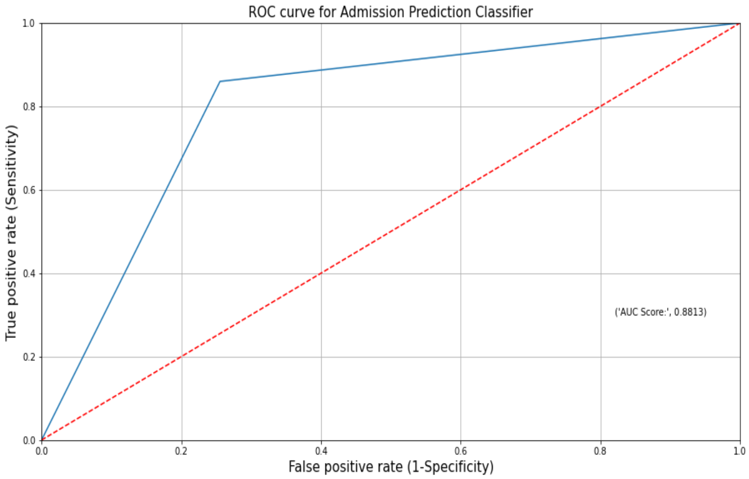


```
              precision    recall  f1-score   support

           0       0.84      0.74      0.79      6166
           1       0.77      0.86      0.81      6166

    accuracy                           0.80     12332
   macro avg       0.81      0.80      0.80     12332
weighted avg       0.81      0.80      0.80     12332
```

Confusion Matrix:



ROC Curve:



HYPER-PARAMETERS:

Pre-pruning can be done by specifying the following hyperparameters:

criterion:

- The criterion available here are gini and entropy
- It determines the basis of a split.

max_depth:

- It is the maximum length of the decision allowed to grow.
- Once the max_depth value is reached the tree will not grow further.

min_samples_split:

- The minimum samples are required to split an internal node.

max_leaf_nodes:

- It helps create the tree with “max_leaf_nodes” in best-first fashion.
- Once the max_leaf_nodes value is reached the leaf nodes will not grow further

n-estimators:

- This is the number of trees you want to build before taking the maximum voting or averages of predictions.

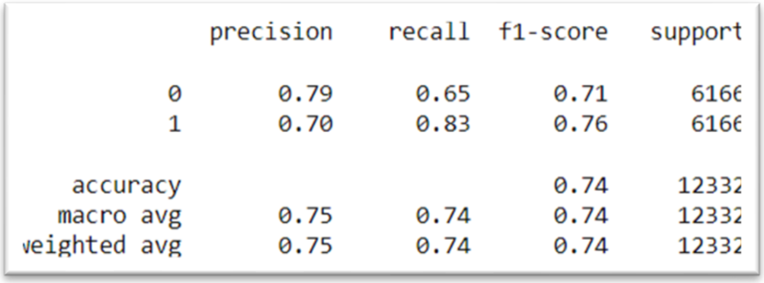
HYPER-PARAMETER TUNING:

- Hyper tuning the parameters is needed to reduce the overfitting of the model and also to further refine the model.
- The hyperparameters can be tuned using the Grid Search method. It considers all the combinations of the hyperparameters and returns the optimal hyperparameter values.
- The tuned parameters resulted from the Grid search method as follows for Decision Tree and Random Forest are as follows:

    - Criterion: Gini
    - Max_depth: 10
    - Min_sample_split: 2
    - max_leaf_nodes: 15

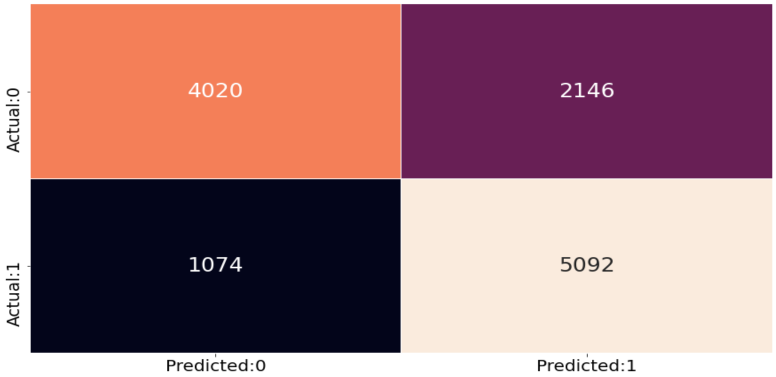
Tunned Decision Tree:

Classification Report:

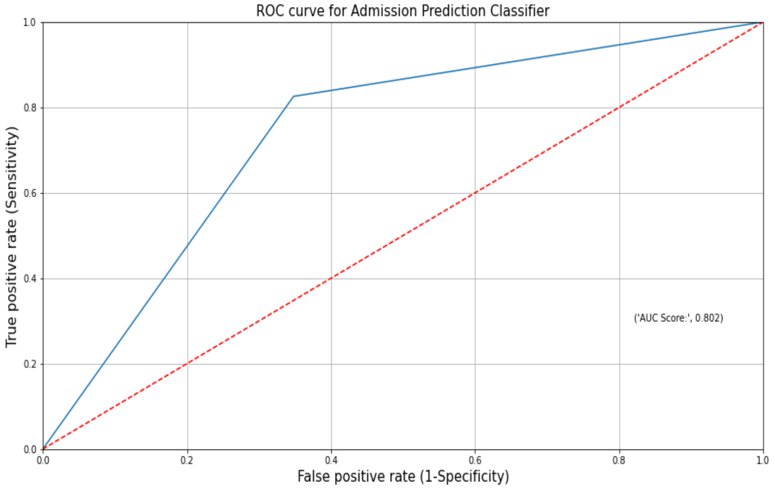


```
              precision    recall  f1-score   support

           0       0.79      0.65      0.71      6166
           1       0.70      0.83      0.76      6166

    accuracy                           0.74     12332
   macro avg       0.75      0.74      0.74     12332
weighted avg       0.75      0.74      0.74     12332
```

Confusion Matrix:

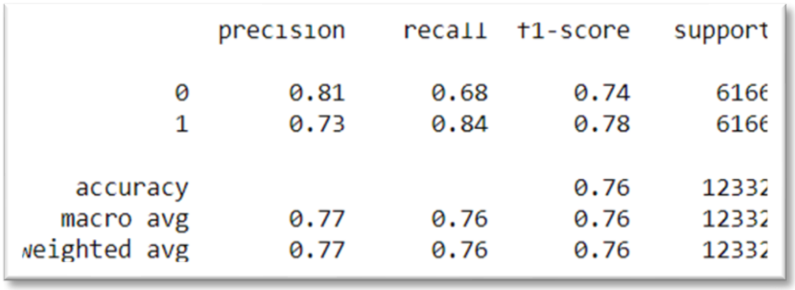


ROC Curve:



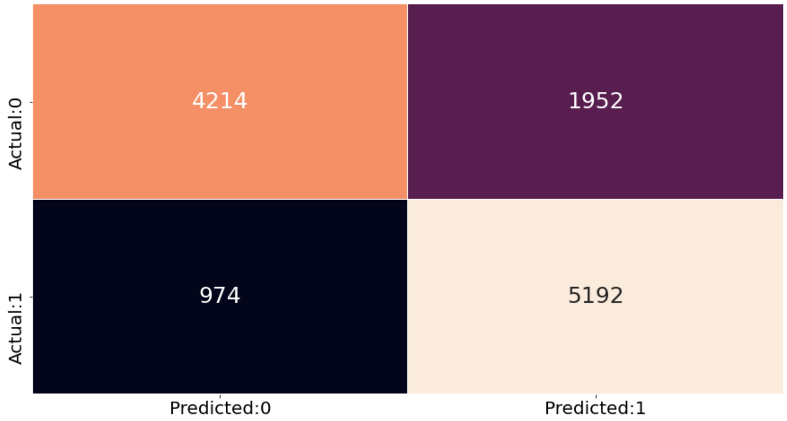
Tunned Random Forest:

Classification Report:

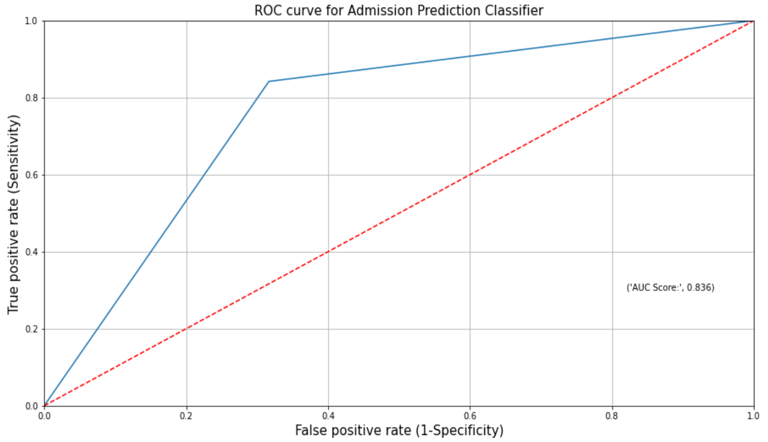


```
              precision    recall  f1-score   support

           0       0.81      0.68      0.74      6166
           1       0.73      0.84      0.78      6166

    accuracy                           0.76     12332
   macro avg       0.77      0.76      0.76     12332
weighted avg       0.77      0.76      0.76     12332
```

Confusion Matrix:



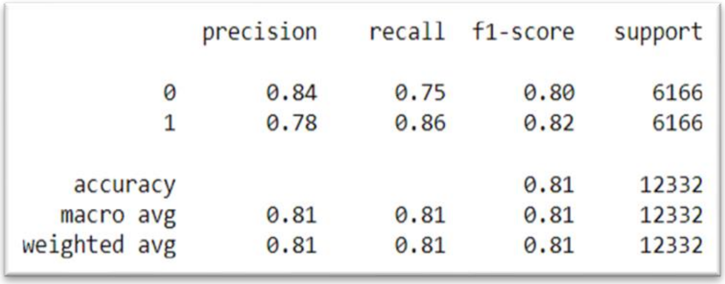
ROC Curve:



Boosting Algorithm: XGBoostClassifier

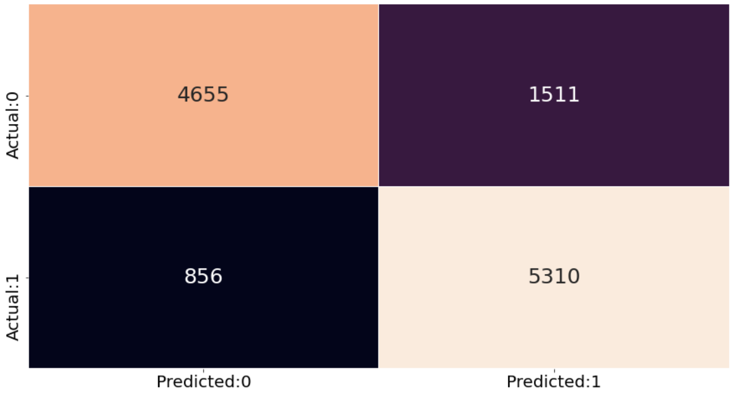
- To improve the performance of the model, boosting classifiers such XGboost classifiers is considered.
- The below are the results obtained from the boosting algorithms.

Classification Report:

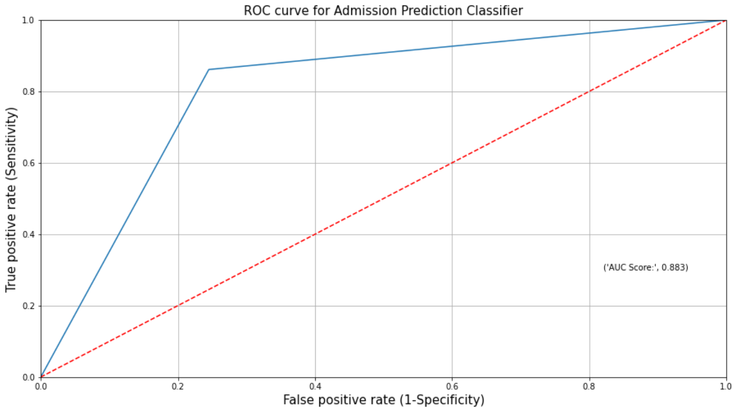


```
              precision    recall  f1-score   support

           0       0.84      0.75      0.80      6166
           1       0.78      0.86      0.82      6166

    accuracy                           0.81     12332
   macro avg       0.81      0.81      0.81     12332
weighted avg       0.81      0.81      0.81     12332
```

Confusion Matrix:



ROC Curve:



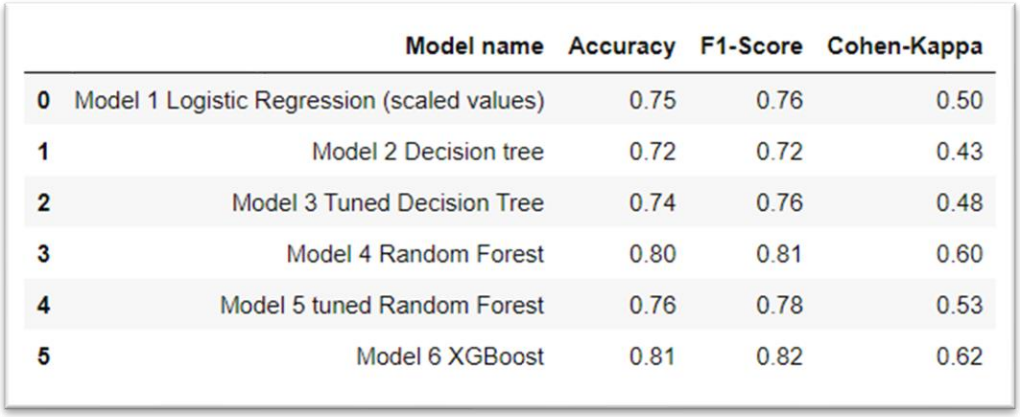
Inferences:

- We started with Logistic regression as our base model which gave us an accuracy of 0.75 and an F1- score of 0.76 but a low Cohen-Kappa value of 0.50. It must also be noted that the data contains a good number of outliers as noticed during EDA and since Logistic Regression is sensitive to outliers it's better to move to models that aren't sensitive to outliers.

- Tree-based models aren't affected by outliers hence we used decision tree and random forest models and further tuned their hyper parameters wherein the performance of the decision tree improved after hyperparameter tuning while tunning of the random forest could not improve the overall model performance. Hence, we considered a boosting algorithm to improve the performance of our model.

- We used XGBoost Classifier that gave us the highest Accuracy, F1-score, and Cohen-Kappa value. The total false negatives and false positives were also the lowest for this model.

MODELS COMPARISON:



	Model name	Accuracy	F1-Score	Cohen-Kappa
0	Model 1 Logistic Regression (scaled values)	0.75	0.76	0.50
1	Model 2 Decision tree	0.72	0.72	0.43
2	Model 3 Tuned Decision Tree	0.74	0.76	0.48
3	Model 4 Random Forest	0.80	0.81	0.60
4	Model 5 tuned Random Forest	0.76	0.78	0.53
5	Model 6 XGBoost	0.81	0.82	0.62

Conclusion:

Despite hyper tuning the parameters of the Random Forest model, the model has not given better results compared to the first random forest model.

The XGBoost model provides better results with an accuracy of 0.81 and an F1-score of 0.82 but the data still contains outliers and this boosting technique is sensitive to outliers.

Since the main requirement is to maximize the Accuracy and F1-score, it's ideal to consider the random forest base model as final model.

PERFORMANCE METRICS:

- **Confusion Matrix:**

  It is the performance measure for the classification problem. It is a table used to compare predicted and actual values of the target variable.

- **ROC:**

  ROC curve is the plot of TPR against the FPR values obtained at all possible threshold values.

PERFORMANCE EVALUATION METRICS:

- **Accuracy:**

  Accuracy is the fraction of predictions that our model got correct. Higher the accuracy of the model better is the model.

- **Precision:**

  Precision is the proportion of positive cases that were correctly predicted.

- **Recall:**

  A recall is the proportion of actual positive cases that were correctly predicted.

- **F1 score:**

  F1score is the harmonic mean of precision and recall values for a classification model.

- **Cohen Kappa score.**

  Kappa statistic is a measure of inter-rater reliability or degree of agreement.

BUSINESS JUSTIFICATION:

- The main motive is to improve the performance of the model. As per the business scenario, the model is predicting a substantial number of songs to be a hit when in reality they are not likely to be hits. In this scenario, if Spotify can predict if a song is likely to be a hit, then that song can be suggested by their system to people listening to songs of the same category or songs by the same artist.
- This could help increase the time spent on the app by the users which can further motivate them towards purchasing premium version of the app.
- Alternatively, even if a user spends more time on the app and still doesn't want to purchase the premium version, Spotify can still generate revenue by playing relevant advertisements in between songs.
- Music production requires the creative flow of the entire team for making a track but despite all their efforts they stand a high chance of failure. A model that is able to predict the likelihood of a song being a hit could help music producers and signers as well. Spotify could partner with people from the industry and share such information to an extent where both, the creators and Spotify, both can benefit from this analysis.

References:

- https://www.kaggle.com/datasets/theoverman/the-spotify-hit-predictor-dataset
- https://scholar.smu.edu/cgi/viewcontent.cgi?article=1204&context=datasciencereview
- http://cs229.stanford.edu/proj2018/report/16.pdf