**ISTM 660 Applied Predictive Analytics**

**Final project report**

By: *Group 1* | Date: *8th May 2023*

*Harsh Chaudhari | Tyler Huffman | Charan Jirra*

**Introduction**

We considered several domains to pick from before searching for a data set (from that domain) that will be used in our project. All the team members had some experience working in Healthcare, Retail, Supply chain and Industrial domain, thus we decided to avoid these areas. Next was music, which we all agreed is benefiting a lot from the recent improvements in Machine Learning (ML) and Artificial Intelligence (AI). One of the most fundamental applications of ML in music is a recommendation engine. As the name suggests, its job is to recommend users new songs that they can listen to based on several factors. This forms a backbone of how users use these popular services. An important task of the recommendation engine [in laymen terms] is to understand a song. This can mean several things, such as giving a song a popularity scale, categorizing it as a genre, create a cluster of similar songs based on their audio features, such as bass, treble, tempo and so on.

This led us to discover GTZAN data set [1] which consists of 1000 audio clips of different songs. This was further processed by team of researchers from the Echo Nest, Columbia University, and UC San Diego. The team included Brian Whitman, Paul Lamere, Tristan Jehan, and Thierry Bertin-Mahieux for Dr. Dianne Cook’s (Monash University) statistics class. It is a comprehensive dataset that includes metadata and audio features of songs, as well as binary indicators for each of the 10 genre labels used to annotate the songs. The metadata features include song title, artist, year of release, and duration, while the audio features include tempo, key, loudness, and spectral properties.

Our raw data (that is downloaded from source) had 59,600 records with 10 Genres (classes). The Genre column will become our target variable (alternatively known as dependent variable or response variable) that we are trying to predict for. To achieve this, we will train multiple classification models and compare the results by testing in on data that is unseen by the models. We finally compare the performance of these models based on a consistent metric. We have decided to use a confusion matrix which consists of a square matrix with rows and columns representing the actual and predicted genres, respectively. The cells of the matrix show the number of songs that were correctly or incorrectly classified into each genre. This will help us assess the performance of each model and help us decide the best model. As a prelude to the analysis, here’s a list of items that you can expect in the upcoming sections:

Dataset analysis and pre-processing: Here, we explore our raw data to get a sense of what we are dealing with. This includes understanding distribution of feature variables, finding missing values, and checking correlations between numerical variables. This will be followed by preparing dataset for the models by addressing issues and challenges that are discovered in the analysis.

Modelling and predictions; Conclusion: In this step, we split the dataset into Training (80%) Testing (10%) and Validation (10%) dataset. The training and testing data will be used to train the models and optimize them. We conclude the analysis by further validating the results of the best model by predicting validation data to confirm if the model is not showing a higher accuracy on just the train/test data but also on completely unseen data as well. This concept is known as bias vs variance. Bias means how good is the model at approximating the true relation between response variable and predictor variable, meanwhile variance refers to the tendency of a model to capture not only the underlying patterns but also the noise in the data leading to a good training performance but a poor validation performance.

*[1] Raw Dataset sources:* [*http://millionsongdataset.com/*](http://millionsongdataset.com/)

Next steps: In this section, we reflect upon our journey and mention what could have improved our analysis and recommend what should be the next best course of action to make the most out of our learnings from the project.

**Part 2**

**Data Set Background:**

As mentioned in the introduction, the data set that our team used for this project is a combined effort of researchers from the Echo Nest, Columbia University, and UC San Diego. The data set consists of a list of audio fingerprints for various songs. Echo Nest got famous for creating a program that does audio fingerprinting. This program takes in audio signals, or songs in this case, and converts them into numeric and categorical variables so that the audio can be represented mathematically and can be stored as a set of feature variables. Some of the variables within the audio fingerprints include metrics for loudness, tempo, duration, and other numeric variables.

The dataset contains a class variable called "genre" which categorizes each song into a specific music genre. The raw data set that we have used contains 10 different classifications for genre throughout the 59,600 different songs that are in the data set. For our analysis we will be using the genre variable as the dependent class variable that we want to predict, and we will use the other numeric and categorical variables as independent variables to help predict the genre of music for each song. A summary of the initial understanding of the dataset is highlighted below:

|  |  |
| --- | --- |
| ***Raw Data Summary*** | |
| **Link to the dataset** | The raw dataset can be found [here](https://drive.google.com/file/d/1EBTIaR-GRFJ3VfLH65X8GKiWYZaH56h7/view?usp=sharing) |
| **Description** | The data set has 34 columns specifying a song ID, labelled genre and so on. Each column’s name is self-descriptive, though some contextual knowledge will be required. [Complete list of variables] |
| **Number of records** | 59,600 |
| **Number of features\*** | 3 Categorical features 26 Numerical features |

*\* Data dictionary available in Appendix*

For this analysis, since we have a data set with multiple different genres, we will have to use multi-class classification to classify songs into the various genres in the dataset. The raw data set that we have used consisted of 59,600 different songs across 10 different genres. The data set also had 34 different variables for each of the different songs, 4 of which were categorical, 26 were numeric, and the rest were identifiers for the song. These identifiers consisted of TrackID, Artist Name and Title of the song.

**Descriptive analysis and pre-processing:**

We ran some descriptive summaries to better understand our dataset and discovered some very interesting findings:

1. *Identify missing values.*
   * We found out that none of the fields have any missing values indicating that the dataset is clean and does not require us to take any actions such as imputation (back filling) or drop records due to this.
2. *Distribution of total number of records by class.*
   * We saw that that the distribution of classes is highly unbalanced. This posed a challenge that will be further discussed in the following section of the report.
3. *Identify outliers/check the distribution of data:*
   * *Numerical Variables:*
     + We looked at the distribution of individual variables and found out that perceived outliers (such as Quartile 3 Average timber coefficient is ~3 and Maximum is 80) is indeed possible and is a quality of audio.
     + Tempo = 0: This should not happen, as no song in its entirety can be silent (that is, have 0 beats per minute).
     + Loudness < 0: This is also possible as loudness is measured in a logarithmic scale, and negative values mean that the sound was softer than the reference value.
     + The dataset contains multiple variables that are measured in different scales. This needs to be addressed in models such as KNN where scaling is important.
   * *Categorical Variables:* 
     + Nothing concerning was discovered for the categorical variables.
4. Check correlation between the numeric variables.
   * We found out that a lot of variables were highly correlated. This would be a problem and thus needed to be considered while preprocessing the data.

*Refer to appendix for detailed statistical summaries for each of these insights.*

Originally, for our models, our team wanted to try to utilize the entire dataset to make predictions about all 10 different genres listed in the dataset. After trying to run some decision trees for the raw dataset and waiting almost ~1 hour for the model to run, we quickly realized that it would not be feasible for the hardware that we had available, so we had to trim the dataset into a form where it was usable on our computers to build the models. Similarly, we encountered a variety of challenges that we had to consider when processing the data set and preparing it for modeling. These include:

* The size of the data set
* The correlations between the different variables
* The imbalance between the different genres
* Songs that had a tempo equal to zero
* Different scales of measurement for different set of variables

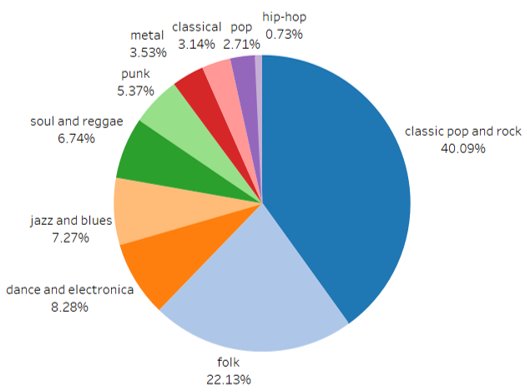
We quickly realized that each of these challenges needs to be addressed before we start to modelling.

To start, the size of the dataset was a very big problem when working with basic hardware and trying to build models using multi-class classification. The size of the dataset was so large that it was impossible to run more time-expensive models on our laptops, such as random forests and support vector machines. Due to this we had to reduce the size of the data set to make it much more manageable.

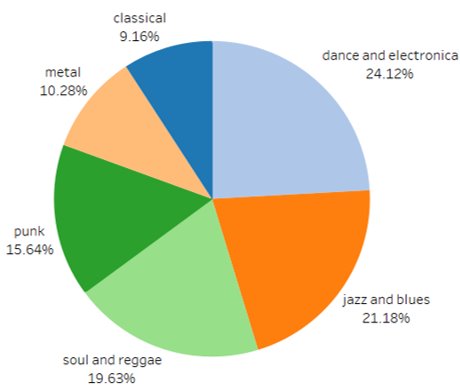
As seen in the descriptive analysis, we observe how heavily the dataset was biased towards certain genres. The initial dataset had over 60% of the entire dataset coming from just two genres. In addition to that, the smallest genre, hip-hop, made up only 0.73% of the overall dataset. While getting a sense of the how much time it will take for use to process the entire dataset, we also realized that due to this imbalance, the initial models that we were able to build only predicted the two genres that made up the majority of the dataset, which was very problematic for the analysis we were trying to do because it lead to very biased models. Due to these issues, we decided to create a sub-sample of the data to use to build our data and resolve two challenges at the same time: Very large dataset and highly imbalanced dataset.

To do that we decided to remove the top two genres in the dataset, classic pop/rock and folk, and the bottom two genres, pop and hip-hop. This change left us with a dataset that was much more manageable. Our dataset after processing consisted of 20,397 different songs across 6 different genres. These genres included dance and electronica, jazz and blues, soul and reggae, punk, metal and classical. Each one of these genres will be included in the multi-class classification that we implemented into each model. Below is a visual representation of what the dataset looks like before and after the processing that we did.

**Class distribution in raw dataset:**



**After Removing records with top 2 majority and 2 minority classes:**

****

In addition to these two adjustments that we made to the data set, we also had to consider the correlation between each variable that we were using to build the model. The initial dataset consisted of 34 different variables for each song. For these variables we ran a correlation plot to identify variables in the dataset that were highly correlated to one another.

What we found was that there were 8 different variables that were highly correlated with one another with around ~70% correlation or more. All other variables that we kept were, at maximum, 40% correlated or less. So, to clean up the dataset and only use significant variables, we decided to remove the 8 variables with a high correlation and remove the 3 identifier variables as well. After removing all those variables our final data set consisted of 22 different independent variables and 1 predictor variable.

***Correlation plot for all the numeric variables:***

A picture containing text, screenshot, pattern, line

Description automatically generated

The last thing that we had to do when processing the dataset was to handle rare songs that were listed with a tempo of 0. When looking at the statistics of the dataset (Mean, Mode, and Range) for each variable, we found that around ~100 different songs within the dataset were listed with a tempo of 0. The tempo of the song is a measure speed of the music, so these songs that were listed with a tempo of 0 technically were not songs because they had no speed to the music or no change in the rhythm of the music. To deal with these records, we simply removed them from our dataset.

After all the data was analyzed and processed, the final pre-processed dataset that we used consisted of 20,397 different songs across 6 different genres of music. These genres included dance and electronica, jazz and blues, soul and reggae, punk, metal and classical. Our processed dataset also had 23 different columns with the genre column being used as the independent variable that we wanted to predict.

As a final step of pre-processing, we normalized the numerical variables, so that the challenge of having different units of measurement get causing vastly different scale of distribution of data can be overcome. This is good practice and will help avoid running into pitfalls of models such as KNNs that are sensitive to predictor variables of different scales.

After processing the data down, before building the models, we decided to **use 80/10/10 split for the data for training, testing, and validation** respectively. This will leave us with 16,320 rows for training, 2,040 for testing and 2,037 for validation. We then ensured that the distribution of the randomly sampled data in proportion that is representative of the overall population so that we are not missing out on any genres. We confirmed the same by looking at the distribution of the number of records in the training set.

**Predictive Models**

After preprocessing the data, the next step is to train, optimize and compare the data. The response variable is categorical and various classification models were implemented on the dataset for prediction. We will be focusing only on Accuracy in this analysis to compare the models. This is critical to do apples-to-apples comparison. While other parameters were looked at, they do not provide any added value to our analysis. An example would be Specificity vs Sensitivity comparisons. We don’t have any preference of true negatives over true positives and thus preferring to look at one over the other will not be particularly helpful.

**Note:** Logistic regression was excluded from our analysis because of two main reasons. First, logistic regression works well when the response variable is binary but our response classes for prediction are 6. Second, it is always difficult and tricky to decide where to put the threshold for a multi-class classification problem.

1. **Random Forest:**

Random Forest (RF) is a popular machine learning technique used for classification and regression analysis. It is based on a decision tree method where multiple trees are produced to predict the output of the response variable using different predictors at each split. It has high accuracy and can handle high-dimensional data.

*Some general things to remember before using Random Forest include: Every feature used to train is relevant to the target variable. RF assumes that the decision boundary is non-linear. The relationship between the features and the target variable is complex and cannot be explained by a linear model. The data is not imbalanced.*

Random Forest classification model is trained with our training dataset with various tuning parameters of number of predictors (mtry) used for splitting each node and the number of trees generated (ntree) for modelling. We used mtry values from 2 to 21 predictors and ntree values of 500, 1000, 1500, 2000, 2500 and 3000. We achieved the highest overall accuracy of 69% at tuning parameters of mtry=21 and ntree=1000.

**Confusion Matrix for test data using Random Forest model (ntree = 1000, mtry = 21, trained on training data):**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Predicted vs Actuals | Classical | Dance and Electronica | Jazz and Blues | Metal | Punk | Soul and Reggae |
| Classical | 78% | 2% | 5% | 0% | 1% | 0% |
| Dance and Electronica | 6% | 65% | 13% | 6% | 10% | 16% |
| Jazz and Blues | 7% | 8% | 68% | 6% | 5% | 8% |
| Metal | 1% | 2% | 1% | 70% | 6% | 0% |
| Punk | 2% | 7% | 3% | 14% | 69% | 3% |
| Soul and Reggae | 6% | 15% | 10% | 4% | 10% | 72% |
| *Total* | *187* | *490* | *433* | *210* | *319* | *401* |

*\* How to read this table: row headers are for predicted classes and column headers are for actual classes in test data. A % value represents the total number of actual classes; how many classes were predicted accurately. i.e., 78% of actual Classical songs were also predicted as Classical by our model, meanwhile it was miss-predicted 6% to Dance and Electronica class. Totals represent the total number of records in test data for each genre.*

As we can see in the confusion matrix the percentage of classical genre correctly classified is 78%, 6% classified as Dance and Electronica, 7% as Jazz and Blue and so on. 65% of Dance and Electronica, 68% of Jazz and Blues, 70% of Metal, 69% of Punk and 72% of Soul and reggae genres data is correctly classified.

1. **K-Nearest Neighbor:**

KNN (K-Nearest Neighbor) is a popular non-parametric supervised machine learning technique used for classification and regression analysis. This technique uses K nearest neighbors from the training data and takes the majority vote among the neighbors to predict the class of the new data. K is the tuning parameter in this technique, smaller the K value higher the model flexibility.

*Some general things to remember before using KNN include: The data is normalized; it is good practice to normalize the data before using KNN because it is sensitive to the feature scales. The data is independent: KNN assumes that the features are independent of each other. This means that the value of one feature should not affect the value of another feature.*

We have implemented this method to our dataset with various values of tuning parameter K ranging from 1 to 100. The best overall accuracy of 63% was achieved at the K value of 13.

**Confusion Matrix for test data using KNN model (K = 13, trained on training data):**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Predicted Vs Actual | Classical | Dance and Electronica | Jazz and Blues | Metal | Punk | Soul and Reggae |
| Classical | 73% | 3% | 9% | 1% | 1% | 1% |
| Dance and Electronica | 6% | 51% | 7% | 4% | 5% | 12% |
| Jazz and Blues | 11% | 16% | 63% | 4% | 6% | 9% |
| Metal | 1% | 4% | 2% | 64% | 11% | 1% |
| Punk | 3% | 7% | 4% | 23% | 63% | 5% |
| Soul and Reggae | 6% | 19% | 15% | 4% | 13% | 73% |
| *Total* | *187* | *490* | *433* | *210* | *319* | *401* |

*\* How to read this table: row headers are for predicted classes and column headers are for actual classes in test data. A % value represents the total number of actual classes; how many classes were predicted accurately. i.e., 73% of actual Classical songs were also predicted as Classical by our model, meanwhile it was miss-predicted 6% to Dance and Electronica class. Totals represent the total number of records in test data for each genre.*

As we can see in the confusion matrix the percentage of classical genre correctly classified is 73%, 6% classified as Dance and Electronica, 11% as Jazz and Blue and so on. 51% of Dance and Electronica, 63% of Jazz and Blues, 64% of Metal, 63% of Punk and 73% of Soul and reggae genres data is correctly classified.

1. **Support Vector Machine:**

Support Vector Machine (SVM) is a supervised machine learning technique which can be used for both regression and classification. This technique tries to divide the plane into several classes and draw a hyperplane separating them. If the data is non separable it converts the data into higher dimensions using kernel functions.

For multi-class classification SVM uses one-to-one and one-to-many techniques. The tuning parameters are kernel function, cost, and gamma. The cost hyperparameter ‘C’ controls the width of the margin. The smaller C value has a wider margin and allows more misclassifications while the larger C value has a narrow margin and allows less misclassifications. The gamma hyperparameter determines the decision boundary. The higher the gamma value the model tries to fit training data more closely leading to possible overfitting while the smaller value of gamma makes the decision boundary more flexible and possible underfitting. It is very important to select the optimum cost and gamma values which can be achieved by using cross validation.

*Some general things to remember before using SVM include: SVM works best when the number of samples in each class are approximately equal. For imbalance data the algorithm favors the majority class. SVM requires feature scaling to be done to all the features in the dataset.*

Kernel functions such as Linear, Radial and Polynomial were used in our analysis. Cost parameters of 0.001, 0.01, 0.1, 1,5,10 and 100 were tested along with gamma values of 0.1,0.5 and 1. The best cost and gamma values were 1 and 0.1 respectively. The overall highest accuracy achieved with the above parameters was ~70%, which is higher than all the models we implemented in our analysis. *[Note: The results of SVM model with Linear and Polynomial kernel are present in the Appendix]*

**Confusion Matrix for test data using SVM Radial model (Cost = 1, Gamma = 0.1, trained on training data):**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Predicted Vs Actual | Classical | Dance and Electronica | Jazz and Blues | Metal | Punk | Soul and Reggae |
| Classical | 77% | 1% | 3% | 0% | 1% | 0% |
| Dance and Electronica | 12% | 70% | 16% | 10% | 10% | 18% |
| Jazz and Blues | 4% | 7% | 67% | 4% | 3% | 8% |
| Metal | 1% | 2% | 1% | 72% | 7% | 0% |
| Punk | 1% | 6% | 3% | 13% | 69% | 4% |
| Soul and Reggae | 5% | 13% | 9% | 1% | 10% | 69% |
| *Total* | *187* | *490* | *433* | *210* | *319* | *401* |

*\* How to read this table: row headers are for predicted classes and column headers are for actual classes in test data. A % value represents the total number of actual classes; how many classes were predicted accurately. i.e., 77% of actual Classical songs were also predicted as Classical by our model, meanwhile it was miss-predicted 12% to Dance and Electronica class. Totals represent the total number of records in test data for each genre.*

As we can see in the confusion matrix the percentage of classical genre correctly classified is 77%, 12% classified as Dance and Electronica, 4% as Jazz and Blue and so on. 70% of Dance and Electronica, 67% of Jazz and Blues, 72% of Metal, 69% of Punk and 69% of Soul and reggae genres data is correctly classified.

**Part 3**

**Conclusion:**

Summarizing the performance of each model:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | ***Accuracy: % of True Positives or in another words: % of Test classes correctly predicted*** | | | | | | |
|  |  | Overall | classical | dance and electronica | jazz and blues | metal | punk | soul and reggae |
| *Optimized Model | Parameters* | *RF | ntree = 1000, mtry = 21* | **69%** | **78%** | **65%** | **68%** | **70%** | **69%** | **72%** |
| *KNN (K=20)* | **63%** | **73%** | **51%** | **63%** | **64%** | **63%** | **73%** |
| *SVC | Cost 1* | **63%** | **74%** | **58%** | **61%** | **67%** | **67%** | **60%** |
| *SVM Radial| Cost 1; Gamma 0.1* | **70%** | **77%** | **70%** | **67%** | **72%** | **69%** | **69%** |
| *SVM Poly| Cost 1; Degree 2* | **68%** | **74%** | **63%** | **68%** | **66%** | **67%** | **70%** |

Purely looking at the accuracy, we decided that the best model to use would be SVM Radial with a cost parameter = 1 and Gamma = 0.1.

To ensure we have not overfitted the data, we would go ahead and run our best model to predict the validation data. The results closely resembled our test results thus strengthening our belief that the analysis we conducted is correct.

**Confusion Matrix for validation data using SVM Radial model (Cost = 1, Gamma = 0.1, trained on training data):**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Predicted Vs Actual | Classical | Dance and Electronica | Jazz and Blues | Metal | Punk | Soul and Reggae |
| Classical | 73% | 1% | 2% | 0% | 0% | 0% |
| Dance and Electronica | 14% | 71% | 15% | 6% | 10% | 15% |
| Jazz and Blues | 9% | 8% | 68% | 3% | 2% | 8% |
| Metal | 1% | 2% | 1% | 72% | 7% | 0% |
| Punk | 1% | 6% | 3% | 18% | 73% | 3% |
| Soul and Reggae | 2% | 13% | 11% | 2% | 8% | 74% |
| *Total* | *186* | *490* | *432* | *209* | *319* | *401* |

*\* How to read this table: row headers are for predicted classes and column headers are for actual classes in test data. A % value represents the total number of actual classes; how many classes were predicted accurately. i.e., 73% of actual Classical songs were also predicted as Classical by our model, meanwhile it was miss-predicted 14% to Dance and Electronica class. Totals represent the total number of records in test data for each genre.*

**Recommendations and Next Steps:**

After conducting our analysis and looking back on what we did, there are several recommendations that we have learned which we would change if we were to do this analysis again. The first, and probably the biggest point of recommendation is to always check the data and analyze the data before even trying to run predictive models. At the beginning of the project, we did not analyze the dataset as much as we should have and, initially, we just tried running the models on the original dataset without much cleaning or processing. This was a huge oversight on our part and it lead to many problems within our initial models. After running into some of these problems, we then when back to the dataset and analyzed it and were able to make the changes that were mentioned in the data analysis section to come up with a dataset that was clean and manageable for our personal laptops. This is one of the biggest problems we encountered when going through the project and if we were to do this again, we would definitely take the time to clean up the dataset before beginning to run the models.

Another recommendation that our team came up with when looking back on our experience is to stay motivated and not give up. This may seem like a general thing when dealing with any project, but our team struggled with this when building our models. Since we were working with multiclass classification, the accuracy of our models were a lot lower then we were expecting. When seeing the bad results, it lowered our confidence and made it very difficult to stay motivated to continue. It took us some time to realize that, with this type of analysis, the accuracy of our models is bound to be much lower than traditional binary classification because it is much harder to predict multiple classes correctly. Additionally, when working with music genres, there is a lot gray area when determining the genre of songs because one song may take influences from multiple different genres. This could lead to the misclassification of a song simply because the genre of the song could be multiple different genres. After realizing these things, it made it much easier to see why our models were coming out with lower accuracies and if we were to do this again, we would keep that in mind.

When going through the project we were limited in what models we could use since we were conducting a multiclass classification project. With this analysis the only models that we could use that we learned in class were KNN, Random Forests, and SVMs. Also, with the hardware that we had; it was very difficult to tune the models because it took such a long time for them to run. Due to this, if we were to continue with this project in the future, we would like to explore other predictive models that we did not discuss in class. One model that, we believe, would be a good one to try is neural networks. We think neural networks would be a good next step because they are often used in building predictive models for multiclass classification because they can process a lot of data very quickly and usually generate high accuracy. On the downside, they take a lot of computational power to run effectively so they are not really feasible to run on personal laptops. Due to this, another next step would be to run the models on hardware that has more computational power. With this, we would be able to run the models on the entire dataset and also be able to include the additional genres that we took out of our analysis as well.

**Appendix**

Data dictionary: green cell is our target variable, Blue cell are identifiers, Red are numerical variables and Yellow are categorical variables.

|  |  |  |
| --- | --- | --- |
| ***Sr.*** | ***Column*** | ***Description*** |
| 1 | genre | Dependent variable: to be used for prediction. Consists of 10 classes in raw dataset |
| 2 | track\_id | Identifier |
| 3 | artist\_name |
| 4 | title |
| 5 | loudness | Numerical variable: Measures how loud the audio is on average, measured in dB |
| 6 | tempo | Numerical variable: Measures the pace the audio is on average, measured in beats per minute |
| 7 | time\_signature | Categorical variable: Indicates the number of beats in each bar or measure of music |
| 8 | key | Categorical variable: A set of notes or pitches that are organized around a central pitch |
| 9 | mode | Categotical variables: A set of pitches or scale used to construct a musical piece |
| 10 | duration | Numerical variable: How long the song is |
| 11 | avg\_timbre1 | Numerical variables: Mel-Frequency Cepstral Coefficients (MFCC) that represent the spectral envelope of a sound |
| 12 | avg\_timbre2 |
| 13 | avg\_timbre3 |
| 14 | avg\_timbre4 |
| 15 | avg\_timbre5 |
| 16 | avg\_timbre6 |
| 17 | avg\_timbre7 |
| 18 | avg\_timbre8 |
| 19 | avg\_timbre9 |
| 20 | avg\_timbre10 |
| 21 | avg\_timbre11 |
| 22 | avg\_timbre12 |
| 23 | var\_timbre1 |
| 24 | var\_timbre2 |
| 25 | var\_timbre3 |
| 26 | var\_timbre4 |
| 27 | var\_timbre5 |
| 28 | var\_timbre6 |
| 29 | var\_timbre7 |
| 30 | var\_timbre8 |
| 31 | var\_timbre9 |
| 32 | var\_timbre10 |
| 33 | var\_timbre11 |
| 34 | var\_timbre12 |

Descriptive statistics of Raw Data: Tempo minimum value can’t be 0.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **# of NAs** | **Mean** | **Minimum value** | **Q1** | **Median** | **Q4** | **Maximum value** |
| **loudness** | No NAs were discovered in the data | -11.5 | -54.4 | -14.1 | -10.5 | -7.6 | 2.9 |
| **tempo** | 122.6 | 0.0 | 97.6 | 120.0 | 142.9 | 260.5 |
| **duration** | 249.4 | 0.8 | 180.9 | 228.7 | 287.8 | 2873.8 |
| **avg\_timbre1** | 41.3 | 1.7 | 37.8 | 42.2 | 45.8 | 58.0 |
| **avg\_timbre2** | -7.8 | -318.7 | -38.2 | -2.5 | 28.6 | 448.4 |
| **avg\_timbre3** | 8.7 | -257.5 | -13.3 | 10.0 | 31.3 | 305.1 |
| **avg\_timbre4** | 0.7 | -120.7 | -9.4 | -0.5 | 9.2 | 197.8 |
| **avg\_timbre5** | -2.1 | -149.9 | -15.9 | -2.0 | 11.3 | 223.5 |
| **avg\_timbre6** | -6.9 | -73.0 | -15.1 | -8.0 | 0.1 | 111.7 |
| **avg\_timbre7** | -4.1 | -165.3 | -12.7 | -3.8 | 5.0 | 133.7 |
| **avg\_timbre8** | -2.1 | -66.7 | -7.0 | -2.2 | 2.5 | 81.8 |
| **avg\_timbre9** | 3.2 | -119.5 | -3.0 | 3.5 | 9.8 | 86.8 |
| **avg\_timbre10** | 1.1 | -38.2 | -3.1 | 0.8 | 5.0 | 83.9 |
| **avg\_timbre11** | -0.2 | -96.3 | -2.7 | -0.1 | 2.5 | 52.9 |
| **avg\_timbre12** | 3.7 | -73.0 | -1.6 | 3.4 | 8.6 | 64.8 |
| **var\_timbre1** | 34.4 | 0.7 | 19.8 | 30.5 | 44.1 | 307.2 |
| **var\_timbre2** | 2401.0 | 112.3 | 1399.9 | 2019.0 | 2922.3 | 47892.9 |
| **var\_timbre3** | 2086.3 | 91.1 | 1234.9 | 1809.3 | 2610.7 | 20131.1 |
| **var\_timbre4** | 1607.7 | 89.9 | 932.6 | 1362.5 | 1968.4 | 25471.0 |
| **var\_timbre5** | 917.2 | 50.3 | 627.1 | 828.2 | 1100.5 | 16049.1 |
| **var\_timbre6** | 915.6 | 30.9 | 539.8 | 786.1 | 1137.9 | 8278.8 |
| **var\_timbre7** | 640.2 | 30.9 | 431.1 | 581.8 | 777.4 | 8044.8 |
| **var\_timbre8** | 519.4 | 30.1 | 335.1 | 456.9 | 624.1 | 7192.2 |
| **var\_timbre9** | 418.4 | 22.9 | 280.4 | 371.0 | 497.9 | 6196.0 |
| **var\_timbre10** | 332.6 | 21.9 | 228.6 | 303.6 | 399.9 | 2707.8 |
| **var\_timbre11** | 296.6 | 20.0 | 183.2 | 256.6 | 361.6 | 4780.3 |
| **var\_timbre12** | 318.4 | 21.2 | 220.1 | 287.6 | 373.9 | 9599.9 |

Distribution of total number of records by class: Heavily unbalanced class distribution

|  |  |  |
| --- | --- | --- |
| Genre | # of total records | Distribution by % of total |
| hip-hop | 434 | 0.73% |
| pop | 1,617 | 2.71% |
| classical | 1,874 | 3.14% |
| metal | 2,103 | 3.53% |
| punk | 3,200 | 5.37% |
| soul and reggae | 4,016 | 6.74% |
| jazz and blues | 4,334 | 7.27% |
| dance and electronica | 4,935 | 8.28% |
| folk | 13,192 | 22.13% |
| classic pop and rock | 23,895 | 40.09% |

Distribution of other categorical variables:

|  |  |  |
| --- | --- | --- |
| **Time Signature** | # of total records | Distribution by % of total |
| 0 | 20 | 0.03% |
| 7 | 1422 | 2.39% |
| 5 | 3294 | 5.53% |
| 3 | 7561 | 12.69% |
| 1 | 9727 | 16.32% |
| 4 | 37576 | 63.05% |

|  |  |  |
| --- | --- | --- |
| **Key** | # of total records | Distribution by % of total |
| 3 | 1913 | 3.21% |
| 8 | 2825 | 4.74% |
| 6 | 3072 | 5.15% |
| 10 | 3924 | 6.58% |
| 1 | 4310 | 7.23% |
| 11 | 4520 | 7.58% |
| 5 | 4615 | 7.74% |
| 4 | 5057 | 8.48% |
| 9 | 6736 | 11.30% |
| 2 | 7158 | 12.01% |
| 0 | 7694 | 12.91% |
| 7 | 7776 | 13.05% |

|  |  |  |
| --- | --- | --- |
| **Mode** | # of total records | Distribution by % of total |
| 0 | 17986 | 30.18% |
| 1 | 41614 | 69.82% |

**Confusion Matrix for validation data using SVM Linear model (Cost = 1, Gamma = 0.1, trained on training data):**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Predicted Vs Actual | Classical | Dance and Electronica | Jazz and Blues | Metal | Punk | Soul and Reggae |
| Classical | 74% | 2% | 6% | 0% | 2% | 0% |
| Dance and Electronica | 9% | 58% | 17% | 7% | 8% | 21% |
| Jazz and Blues | 7% | 11% | 61% | 2% | 5% | 11% |
| Metal | 2% | 4% | 1% | 67% | 7% | 0% |
| Punk | 1% | 6% | 3% | 19% | 67% | 7% |
| Soul and Reggae | 7% | 19% | 13% | 5% | 11% | 60% |
| *Total* | *187* | *490* | *433* | *210* | *319* | *401* |

*\* How to read this table: row headers are for predicted classes and column headers are for actual classes in test data. A % value represents the total number of actual classes; how many classes were predicted accurately. i.e., 73% of actual Classical songs were also predicted as Classical by our model, meanwhile it was miss-predicted 14% to Dance and Electronica class. Totals represent the total number of records in test data for each genre.*

**Confusion Matrix for validation data using SVM Polynomial model (Cost = 1, Gamma = 0.1, trained on training data):**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Predicted Vs Actual | Classical | Dance and Electronica | Jazz and Blues | Metal | Punk | Soul and Reggae |
| Classical | 74% | 2% | 5% | 0% | 1% | 0% |
| Dance and Electronica | 9% | 63% | 11% | 6% | 11% | 15% |
| Jazz and Blues | 9% | 11% | 68% | 5% | 5% | 10% |
| Metal | 1% | 2% | 0% | 66% | 5% | 0% |
| Punk | 2% | 4% | 3% | 19% | 67% | 4% |
| Soul and Reggae | 5% | 17% | 13% | 3% | 12% | 70% |
| Total | *187* | *490* | *433* | *210* | *319* | *401* |

*\* How to read this table: row headers are for predicted classes and column headers are for actual classes in test data. A % value represents the total number of actual classes; how many classes were predicted accurately. i.e., 73% of actual Classical songs were also predicted as Classical by our model, meanwhile it was miss-predicted 14% to Dance and Electronica class. Totals represent the total number of records in test data for each genre.*