Comparision of Different Machine Learning Algorithm for Predicting Music Composer

Ritik Ranjan
Computer Science Student
JK Lakshmipat University

GitHub Repository

Music Composer Prediction

Dataset Link

Google Drive

Abstract

This paper uses MFCCs for learning to compares four different machine learning algorithm for classifying seven different music composer from history. The following machine learning algorithm are used to classify i.e Multinomial Naive Bayes, Logistic Regression, Random Forest Classifier, K Nearest Neighbour, in which K Nearest Neighbour tends out perform other classifier by an accuracy of 87.94%.

Introduction

Music composer classification can help the music enthusiast or even industry to identify that the music was really the publisher composition or it was stolen, or can help when someone don't know the real composer and they want to classify which composer has the high chances of making the composition for given them credit. In the end people can appreciate the composer for making such beautiful art.

In the former ages people used to identify the music composer by the like of the composer tune as unique most composer plays music in the unique way. Like Ludwig van Beethoven has the unique style of joining melodic and non-melodic rhythms in using single chord or so, also his melody section tends to be in the higher octave then usual. Like wise most of the classical, romatic and other type of composer music. As most of the artist have something in common in the compositions they make, that similarity can help, to classify them that, particular composition belongs to some particular composer which in case is MFCC.

Related work

This chapter gives details about the other related project that were working on the same field of classifying music in some sense, and are highly related to this paper.

Jayen Ram & Daniel Salz[2] proposed a paper on perdicting music genre using frequency and lyrics of the songs. The approach uses FFT for short time period which is SFFT, with dataset of 55,000 different songs. In which they also used Neural Network(deep learning) in addition to solely machine learning algorithm. Which gave them 93.2% accuracy on test dataset with SVM and 88.7% accuracy with 100 Neuroned Neural network.

Aarti Singh, Rutvij Gadhiya & Aditya Sawant[3] perposed a paper in 2019 on Music Genre Classification and Retrieval using MFCC. This paper used GTZAN dataset which is one of the largest dataset available on the internet for music related data. This paper used SVM, KNN and HMM(Hidden Markov Model) for classification. Out of which SVM got the accuracy of 56 – 60% on each genre while 95% on classical genre as stated in the paper.

Creme M, Burlin C. & Lenain R[1] proposed a paper in 2016 where they also used MFCC to determine the genre of music. This paper used eleven genre to work on. After which they have used Support vector machine, Neural Network for classifying genre they also used many other algorithm. Out of which they have concluded that Neural network had performed the best out of all the algorithms.

Gathering Data / Feature Selection

This paper used the dataset from Kaggle dataset to retrieve songs(in wav format) for the different composer. Number of composer used in this project is 1 seven, and the composers are Bach, Beethoven, Brahms, Dvorak, Haydn, Mozart, Schubert. The dataset contains 83 songs.

Next step would be convert the audio format (wav) which is raw data into something, so that we can feed into the machine learning algorithm. This paper used MFCC (Mel Frequency Cepstral Co-efficient) to extract audio data into number of coefficient. For each songs MFCC was calculated by the following method: first the song was segmented into ten parts then for every segment the MFCC was calculated according the hop length and for every in between hop thirteen coefficient were calculated.

To give a view every songs would have segment (10 array) for which a single would be number of array (depend upon the number of hops), and inside single array would have 13 coefficient for MFCC. After getting all the required array and corresponding labels then it had been converted the CSV format for easiness, for this we simply get the the 13 coefficient and its label and mapped it as a row in CSV file. A short introduction of MFCC is after this paragraph.

The concept of MFCC (Mel Frequency Cepstral Co-efficient) is was introduced by Davis and Mermelstein in 1980’s and now MFCC is widely used for audio analysis. A single MFC is the representation of short term power spectrum of a sound, which is based on the linear cosine transformation of a log power spectrum on a linear mel scale frequency.

Below is the 3D plot for the first 300 row from the shuffled data for the first 3 coefficient of MFCC.

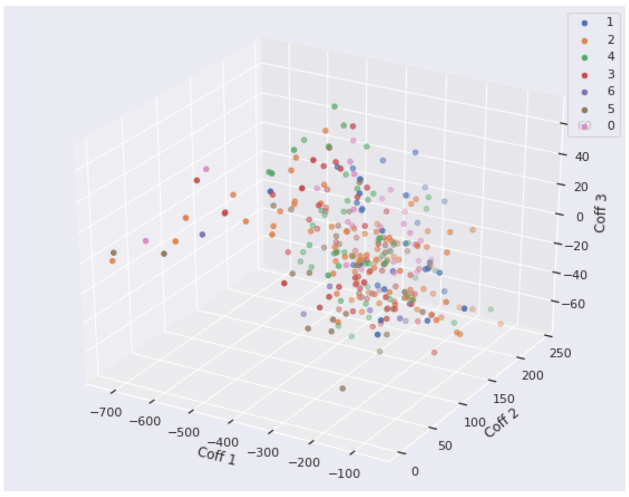


Figure 1: 3D plot of the Dataset

Methods

Logistic Regression

Logistic regression is one of the most simple type of regression technique which uses a sigmoid function to predict the probability of the outcome. Logistic regression is most used to predict binary outcome e.g yes or no. Further it can be extended for multi-class prediction which is also called multi-class classification.

For logistic model we assume that there is a linear relation between the log-odds and predictor variable. In mathematical form we can write the multi variable dependent linear relationship as:

$$\log_e\left(\frac{p}{p-1}\right)=\beta_0+\beta_1 x_1+\beta_2 x_2$$

Here, $p=P(Y=1)$ and $\log_e\left(\frac{p}{p-1}\right)$ is called log odds, by doing simple mathematical manipulation we can retrieve the probality function as

$$p=\frac{b^{\beta_0+\beta_1 x_1+\beta_2 x_2}}{1+\beta_0+\beta_1 x_1+\beta_2 x_2}=S_e(\beta_0+\beta_1 x_1+\beta_2 x_2)$$

where S_e is the sigmoid function. For getting the the values of β_i from the data, one must have to do regression.

For our model training we first scale each data point as

$$z=\frac{(x-u)}{s}$$

where z is the scaled data point , x = original data point, u is the mean of the similar(column) data points and s is the standard deviation of the similar(column) data points.

Linear SVC

This is derived from the very common model named support vector machine linear model. The difference between the two models(sklearn) is that one linear SVC penalizes the intercept and SVM linear not. Also the linear SVC model is implement on liblinear rather then libsvm which provides the linear SVC speed and flexibility of penalizes the intercept.

SVM work by finding the best fit margin the the dataset to separate the dataset classes.

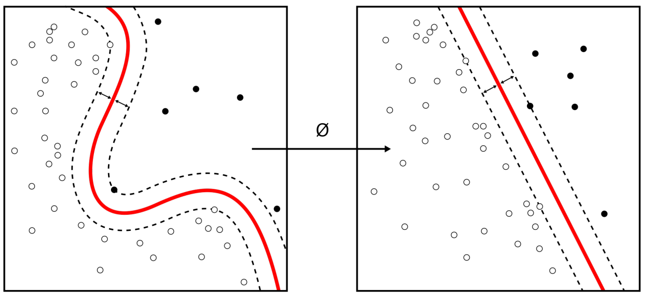


Figure 2: Scatterplot featuring a linear support vector machine's decision boundary (source: wikipedia)

This is done by picking a loss function and then minimizing the loss using the dataset. SVM do this picking a specific choice for the loss function and then minimizing the loss function for the whole dataset for example we pick Hinge loss. Hinge loss for single example look is follows:

$$l(x)=max\{0,\ 1 - x\}$$

Then the cost function can be straightforwardly computed from here. Running a gradient descent and we can minimize the cost function.

K Nearest Neighbour

K Nearest Neighbour is one of the most basic algorithm in machine learning. This algorithm works by computing distance between point, which means this algorithm's performance and accuracy can vastly differ, when switching scaling method or normalizing the dataset.

In KNN, K is a user defined constant, and while classifying a unlabeled point is classified by assigning the label which is most frequent among the K labeled points nearest to the unlabeled point.

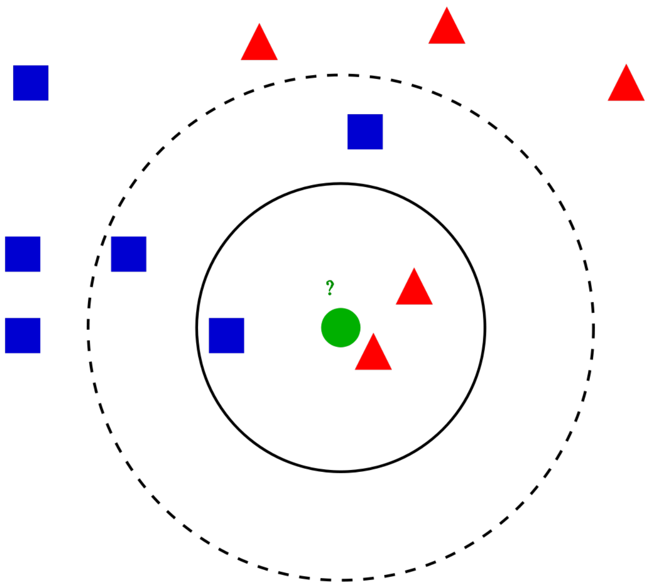


Figure 3: Classifying regions for KNN(source wikipedia)

The intuition of the model is that while predicting, identify which training dataset point is nearest from the predicting points.

In the figure 3, When K=3 (inner solid circle) is the classified for triangle, meaning when a unlabeled data point falls inside the solid circle it while always classified as triangle. When K=5 (outer dashed circle) is the classified for square because square is the most frequent shape inside the dashed circle. This is the way on how KNN works.

While KNN is a very simple and intuitive algorithm, but it is also a very heavy on computing algorithm because it has to compute distance between every other points which scale very much with the size of the data.

Random Forest Classifier

Random Forest classifier also called as random decision forests are the derivative of decision trees. This algorithm can used for both regression and classification, but out interest was only on classification. In simple term what is the most frequent predicted label of the tree is considered as predicted label this is how random forest classifier works.

Working: Let dataset contains total number of k features, the model will choose l features out of k features, where $l < k$. Now after selecting k features model will select a node with highest feature gain and will make it as a root node. Then the model will split the remaining node into child node then repeat this process many times. Finally the model aggregate all the sub tree make into a forest this is also called bagging.

After training, prediction for unlabeled(x) sample can be made by averaging the prediction from all the individual trees inside the forest.

$$\hat{f} = 1/K \sum_{k=1}^{K} f_b(x)$$

where K is the total number of number of features at every node.

Composer Classification

After extracting the feature set from from the music file in form 13 MFCC coefficient for each songs. Then after this and before feeding the dataset to different model, we standardized the data using StandardScaler function from sklearn library. StandardScaler function, which does the following to every data point in the datasets

$$z = \frac{(x-u)}{s}$$

This standardization also helped the model in predicting better especially KNN model, because of the fact it work physical unit.

Result

Using 13 dimension feature space to represent the songs, the KNN model gave the highest followed by the Random Forest classifier. The accuracy given by KNN model is 87.49% on test dataset while Random forest classifier gave 82.72%.

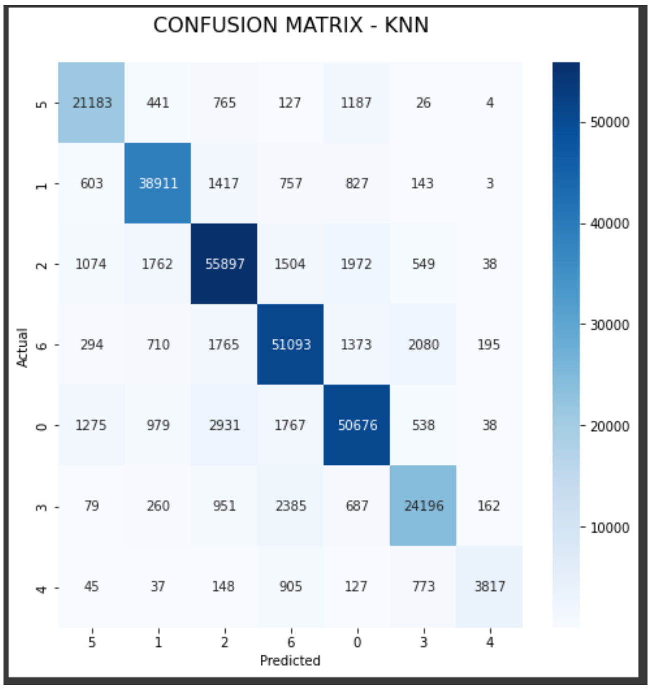
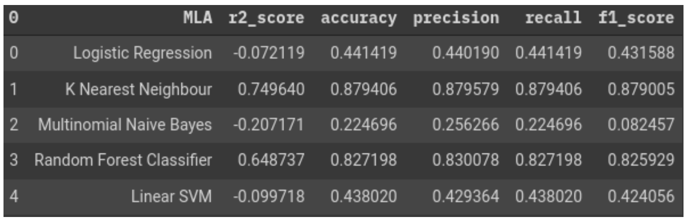


Figure 4: Confusion matrix for KNN

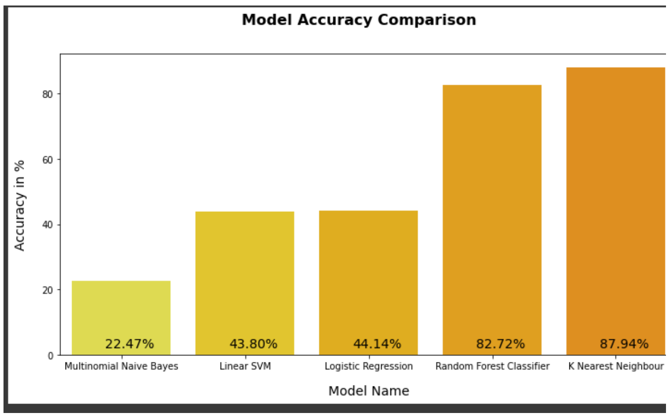
The above diagram is confusion matrix for KNN test dataset. Where 0 is Bach, 1 is Brahms, 2 is Beethoven, 3 is Mozart, 4 is Schubert, 5 is Dvorak, 6 is Haydn.



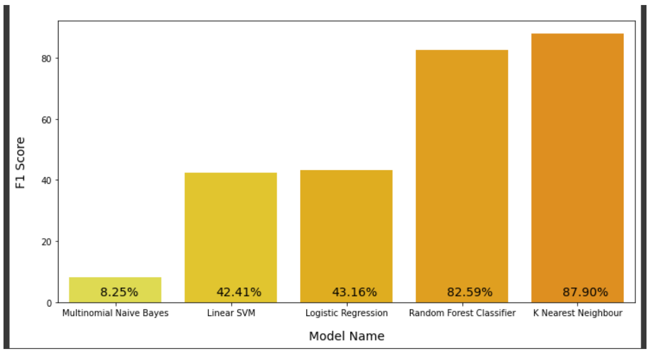
0	MLA	r2_score	accuracy	precision	recall	f1_score
0	Logistic Regression	-0.072119	0.441419	0.440190	0.441419	0.431588
1	K Nearest Neighbour	0.749640	0.879406	0.879579	0.879406	0.879005
2	Multinomial Naive Bayes	-0.207171	0.224696	0.256266	0.224696	0.082457
3	Random Forest Classifier	0.648737	0.827198	0.830078	0.827198	0.825929
4	Linear SVM	-0.099718	0.438020	0.429364	0.438020	0.424056

Figure 5: Machine learning algorithm scores

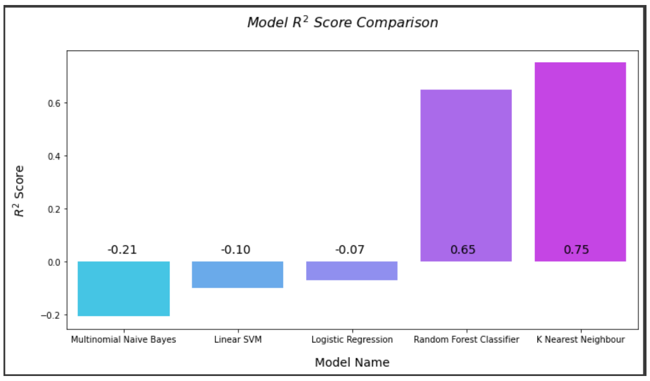
Accuracy Score Comparision



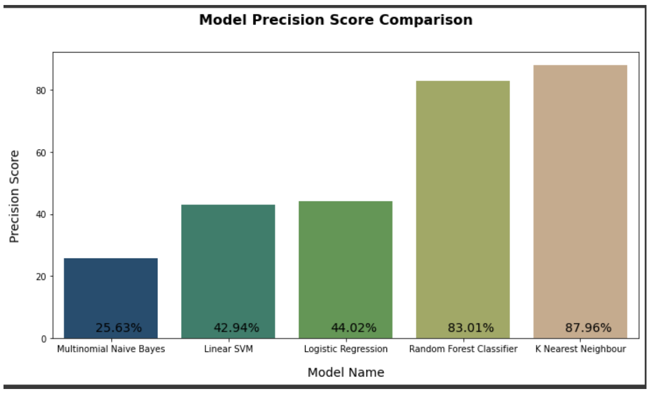
F1 Score Comparision (weighted)



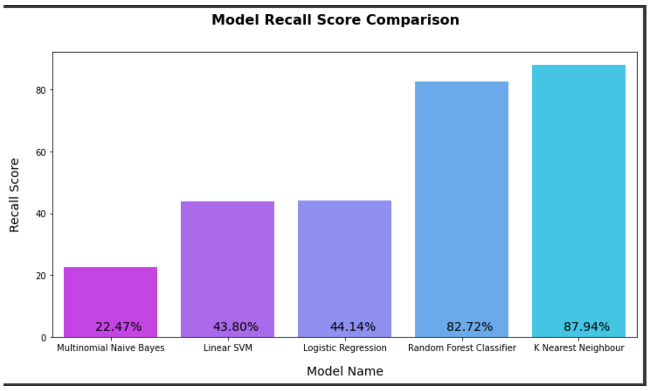
R^2 Score Comparison



Precision Score Comparision (weighted)



Recall Score Comparision (weighted)



Discussion

This paper has successfully implemented a particular piece of software which can be use to recommend of suggest the real author of the composition of the song. We observed that KNN with StandardScaler(pre-processing) was the best method among the other 4 methods. Which is explained by the fact that KNN was better if data is normalised or scaled, but to surprise KNN will work this better then other models.

While doing the project, out of many limitation number of composer in the dataset and training time is one the biggest limitation to the project, else wise the predictions are not too bad.

With more time in hand, we would like test our model with other datasets, and also would like use other famous methods involving deep learning methods such as convolutional neural network and other similar technique. For this we only used wav file as a start, so in future we would like explore other file format for music such as midi.

Reference

1. Creme, M., Burlin, C. and Lenain, R. (2016). *Music Genre Classification*. [online] [Accessed 19 Nov. 2021].

2. McFee, B., Metsai, A., McVicar, M., Balke, S., Thomé, C., Raffel, C., Zalkow, F., Malek, A., Dana, Lee, K., Nieto, O., Ellis, D., Mason, J., Battenberg, E., Seyfarth, S., Yamamoto, R., viktorandreevichmorozov, Choi, K., Moore, J. and Bittner, R. (2021). *librosa/librosa: 0.8.1rc2*. [online].

3. Pedregosa, F., Pedregosa@inria, F., Fr, Org, G., Michel, V., Fr, B., Grisel, O., Grisel@ensta, O., Blondel, M., Prettenhofer, P., Weiss, R., Com, V., Vanderplas, J., Com, A., Cournapeau, D., Varoquaux, G., Gramfort, A., Thirion, B., Dubourg, V. and Passos, A. (2011). Scikit-learn: Machine Learning in Python Gaël Varoquaux Bertrand Thirion Vincent Dubourg Alexandre Passos PEDREGOSA, VAROQUAUX, GRAMFORT ET AL. Matthieu Perrot Edouard Duchesnay. *Journal of Machine Learning Research*, [online] 12, pp.2825–2830.

4. Ram, J. and Salz, D. (n.d.). *Using Song Lyrics and Frequency to Predict Genre*. [online] [Accessed 18 Nov. 2021].

5. Singh, A., Gadhiya, R. and Sawant, A. (2019). Music Genre Classification and Retrieval using MFCC. *IJSRD - International Journal for Scientific Research & Development|*, [online] [Accessed 18 Nov. 2021].