Predicting Musical Eras of Songs

Yandi Li, Jiaqi Xue, Yueyao Zhu

Abstract

The goal of the project is to predict the year in which a certain piece of music was created. We used a subset of Million Song Dataset written for standardized tests to train different models including Naive Bayes Classifier, Generalized Linear Model, Random Forest, and Gradient Boosting Machine. Comparisons are made based on the results(mean square error and feature importance) from various training sizes and different models.

1 Introduction

There are many ways to categorize music, such as genre, artists, target listeners and cultural context. In addition, different musical eras can also be regarded as an important standard to summarize a music feature. Certain music has very obvious mark of their time, and people have special taste for certain types of music as a result.

The goal of the project is to predict the year in which a certain piece of music was created. There are songs that do not have the mark of the time they really belong to. What we try to do, is satisfying the need of people, who have particular favor for a type of music from a certain musical era, for example, music in late 80s. We can give out the list of music that have the common characteristics belonging to that period no matter it was created at that time or not.

2 Dataset

The dataset we are planing to utilize in our project, to design, implement, tune and test our machine learning model is the Million Song Dataset, which is a dataset containing audio features and metadata for a million popular music tracks (http://labrosa.ee.columbia.edu/millionsong/), collected by Columbia University and the Echo Nest. A subset of the dataset exists which has been split into train and test sets that ensures no song from the same artist exist in both the train and test set to avoid the 'producer effect'. The mentioned subset dataset contains 90 attributes, with a most important value "year" ranging from 1922 to 2011. Features that we're planning to explore and utilize to predict the "feeling of the year" includes the timbre average and timbre covariance values. In addition, from the original Million Song Dataset, more metadata about an entry (music) can be extracted such as the genre tags, the artist name, and also evaluations includes the song's danceability, pitches. They features contain discrete data as well as continuous data, that can be used to describe the feeling of the song, of a particular year.

3 Methods

3.1 Baseline Model: Naive Bayes Classifier

Naive Bayes was introduced under a different name into the text retrieval community in the early 1960s[1]. It remains a popular baseline method for text categorization with word frequencies as the features. Considering we are trying to solve a similar problem: categorizing the music into eras based on their musical features, it is reasonable and feasible to choose Naive Bayes classifier as the baseline model for the project.

The assumptions on distributions of features are called the event model of the Naive Bayes classifier. For discrete features like the ones encountered in document classification (including spam filtering), Multinomial and Bernoulli distributions are most popular. Considering the particular characteristics of musical tracks, the team chose Multinomial Naive Bayes model to train the dataset collected from Million Song Dataset.

3.2 Generalized Linear Model

Generalized linear model (GLM) is a flexible generalization of ordinary linear regression that allows for response variables that have error distribution models other than a normal distribution. The GLM generalizes linear regression by allowing the linear model to be related to the response variable via a link function and by allowing the magnitude of the variance of each measurement to be a function of its predicted value. In our project, GLM is a logical choice to try for several reasons. Firstly, we have a large datasets, the scalability of our model is an important factor and GLM is apparently very cheap to compute. Secondly, there are 90 attributes in our data. Choosing the right feature is a difficult task and GLM could solve this by giving corresponding parameter 0. Thirdly, linear regression is reasonable by intuition as features of a song should not have more sophisticated relationship with its era, such as square, log and so on.

3.3 Random Forest

Random forests uses an ensemble of decision trees to make regression and classification models. The trees are built from a training dataset and can be used to make predictions on a test dataset. Random Forests grows many classification trees. To classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification. The forest chooses the classification having the most votes (over all the trees in the forest). Because decision tree learning is invariant under scaling and various other transformations of feature values, it is robust to inclusion of irrelevant features, and produces inspectable models. The group decided to model with random forests learning considering that we have a rather large bases with a handful of input variables and random forests is able to handle thousands of input variables without variable deletion and gives estimates of what variables are important in the classification.

3.4 Gradient Boost Model

Gradient boosting is tried on our dataset as a way of experimenting advanced and more complicated model to see if this can help achieving a better machine learning result. The Gradient Boosting technique works as other boosting methods that combines sub optimized models to form a hopefully better model. As in the scope of machine learning, the methods boost the performance by combining weak learns into one strong learner. At each stage, the model generates an imperfect model, i.e., a weak model that can be efficiently achieved. Then the Gradient Boost Model updates the loss as the difference between the prediction results

and the learning goal. By learning with gradient descent algorithm and generating and combing these "weak" models, the gradient boosting method may be able to generate a model with a good performance.

4 Results

4.1 Baseline Model: Naive Bayes Classifier

After tuning around the hyper-parameters in the Multinomial Naive Bayes model, the following configuration was chosen for the Naive Bayes model used to get the baseline results. Laplace smoothing was not applied in the model's training and predicting process as it turned out that enabling Laplace smoothing didn't yield much improvement of the model. The minimum standard deviation was capped to be at least 0.001 for observations with not enough data, and the minimum probability for observations with not enough data was capped to be at least 0.001 as well.

With the above configuration of the Multinomial Bayes model, a first run with 1000 training entries and 200 testing entries generated a MSE=0.5599 and logloss=11.2997 on the training set and MSE=0.9537 and logloss=25.0771 on the testing set. Another run on a larger training set (10000 data entries) and the same testing set generated MSE=0.91 and logloss=13.07 on the training set and MSE=0.94 and logloss=14.84 on the testing set. Therefore, we have already observed that an increased training set can lead to more accurate result even with the Naive Bayes Model.

4.2 Generalized Linear Model

In the model we built, we included intercept term. Considering both speed and performance, we set objective epsilon to be 0.00001, meaning the model converges if it changes less that this value, and the max iterations to be 50. For testing of our model, we decided to use k-folds validation and set k to be 10.

For the "year" we tried to predict, there are two data type options for us, enum or numeric. At first thought, numeric data seems to be more logical, as time is a continuous variable. However, there are several problems in using this data type.

1. In our training data, each song only has its year of its releasing. So the data is not very accurate if we consider it to be continuous variable.

2. The evolving of music is not linear to time. In some year, there might be big changes in the entire music work while other years seem to be very calm. Therefore even though time is continuous, it does not help to make our model more accurate.

3. The prediction result may not make sense if we use numeric data type. For instance, if there is not song in year 1984, it doesn't make any sense when we predict any song to be released that time. ...

In conclusion, even the variable we try to predict is continuous, it still should be considered a classification problem. By training our data, we now there are different types of music, and then we try to categorize songs into these types in terms of the year they seem to be released in.

Data Type	log-like	objective	MSE	r2	mean_residual_dev	residual_dev	null_dev	AIC
Enum	316563	63.3	88.7	0.298997	88.759805	443799	633091	36803
Num	326122	65.2	91.7	0.296278	91.799219	458996	652240	36971

After deciding the parameters, we used different amount of data to see the performance of our model. We used data from 10000 training samples to 50000 training samples and get result as below.

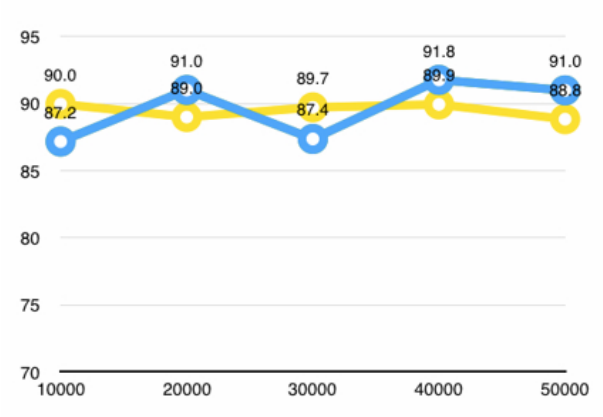
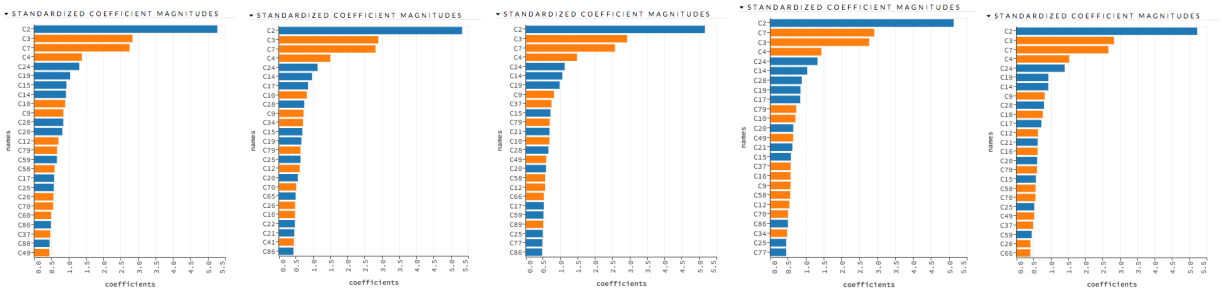


Figure 1: MSE vs. Sample size



(a) 10k samples (b) 20k samples (c) 30k samples (d) 40k samples (e) 50k samples

Figure 2: Features for different sizes of samples

4.3 Random Forest

For the purpose of the project, we choose the number of trees(ntrees) to 50. The mtries is set to be the square root of the number of predictors as recommended. Row sample rate is set to 0.632 after some testing and tuning.

Below is the result get from 10000 to 50000 training sets.

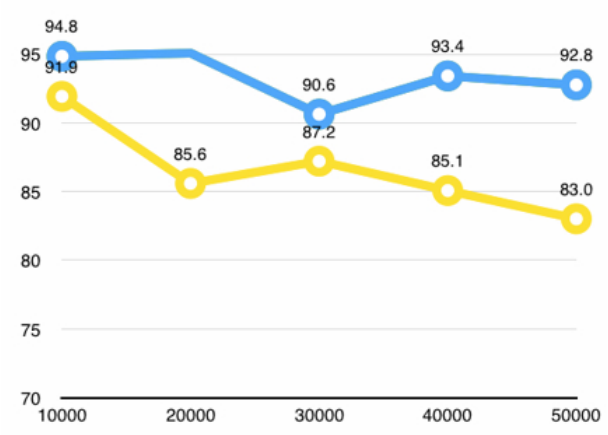
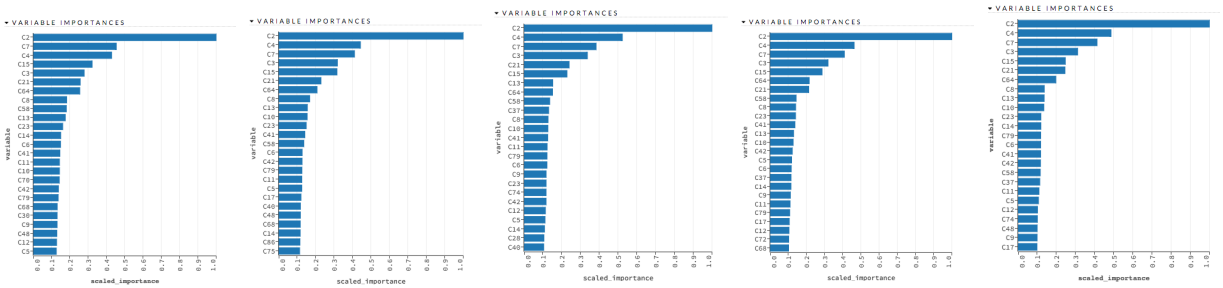


Figure 3: MSE vs. Sample size



(a) 10k samples (b) 20k samples (c) 30k samples (d) 40k samples (e) 50k samples

Figure 4: Features for different sizes of samples

4.4 Gradient Boost Model

As for some major parameters for the gradient boost model used in this project, we choose the number of trees to be 50 with the maximum depth to be 5. The learning rate is set to be 0.1. The r2 stopping threshold is set to be 0.9999 which stops making trees when the R^2 metric exceeds 0.9999. The relative tolerance for metric-based is set to be 0.001 with MSE as the metric.

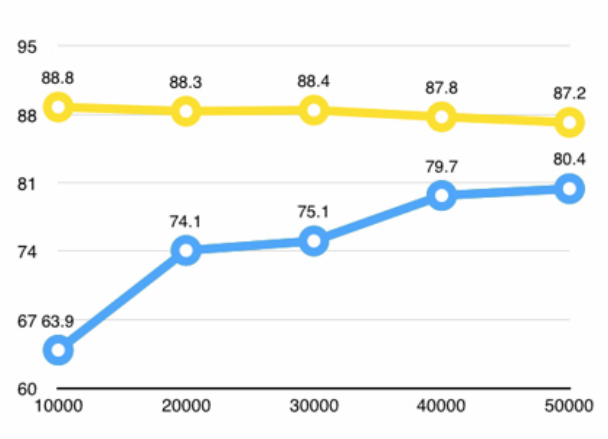
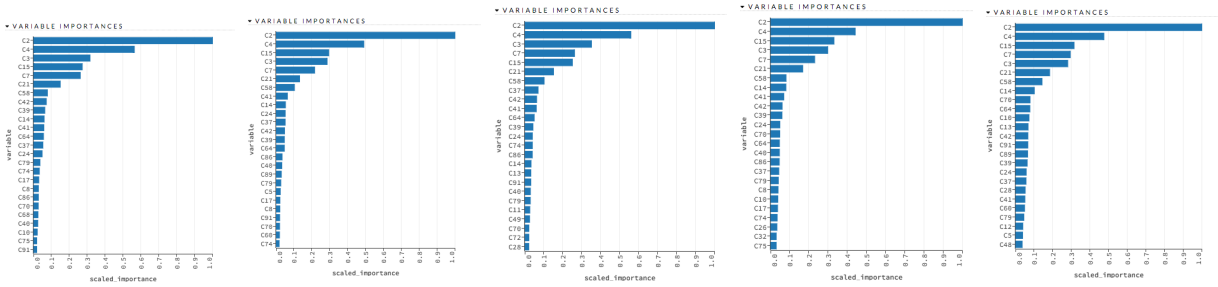


Figure 5: MSE vs. Sample size



(a) 10k samples (b) 20k samples (c) 30k samples (d) 40k samples (e) 50k samples

Figure 6: Features for different sizes of samples

5 Conclusion

5.1 MSE Range

As in the projects, we first observed the MSE from each model to evaluate and compare between the results of different models. Without a huge dataset, all models (GLM, RF and GBM) can generate a MSE that we believe is acceptable for the purpose of our application, as to grab the feeling of years of a music. The task itself is a vague task as human beings can find it extremely hard to tell, and most likely not care which year exactly a song is composed. On the contrary, a user of this application may care much more about whether a group of songs that our model generates share the same feelings, i.e., share the same features.

5.2 Feature Importance

The feature weights generated from each model tends to assign importance to a similar selection of features. This means that each model in our case agrees on which feature leads to the decision of which year a song is created and thus aligns with our expectation of the users enjoying the services from this application.

6 Reference

[1] Russell, Stuart; Norvig, Peter (2003) [1995]. Artificial Intelligence: A Modern Approach (2nd ed.). Prentice Hall.ISBN 978-0137903955.