RELEASE YEAR PREDICTION FROM MILLION SONGS DATASET

Krishna Bharadwaj
Chirag Rao

Contents

1 Introduction

We would like to work on a subset of the million songs dataset which is a freely available dataset from UCI repository(https://archive.ics.uci.edu/ml/datasets/yearpredictionmsd) which consists of multivariate data. The dataset, approximately 1.8 Gb consists of 515345 instances of 90 variables. The variables we will focus on are the timbre average and timber covariance variables in order to predict the release year of a particular song with the help of classifiers such as Support Vector Machines, Logistic Regression with ridge and Random Forest. We will then compare there models with the baseline and tabulate the results.

1.1 Purpose

The purpose of the document is to define the design aspects and the approach taken to predict the release year of songs from the million songs dataset.

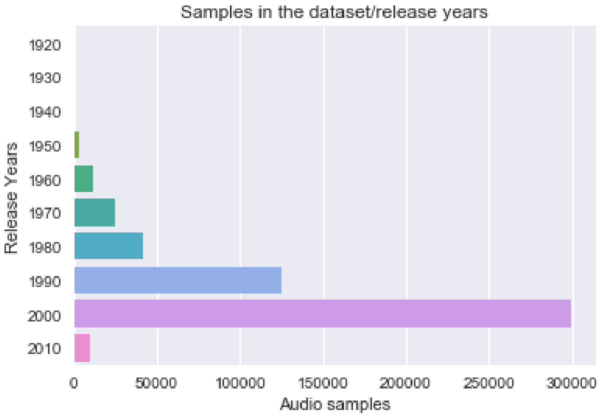
1.2 Related Work

2 Approach

2.1 Data Preprocessing and feature engineering

Since Release year is a continuous variable and we have chosen to go with classifiers for our problem, we have to create classes for the labels. We achieve this by clubbing the songs into decades. As a result, years from 1950 to 1960 would form a single class and so on.

Also, upon visualization of the data, we see that the maximum number of songs in the dataset falls within 2000 and 2010 and there are barely any songs < 1950. Hence, we will also be grouping all the songs that were released before 1950 into a single label.



2.2 Metrics

Since we are going with the classifier approach, we will be using the accuracy of each of the model to predict the release decade to evaluate the different models that are going to be trained

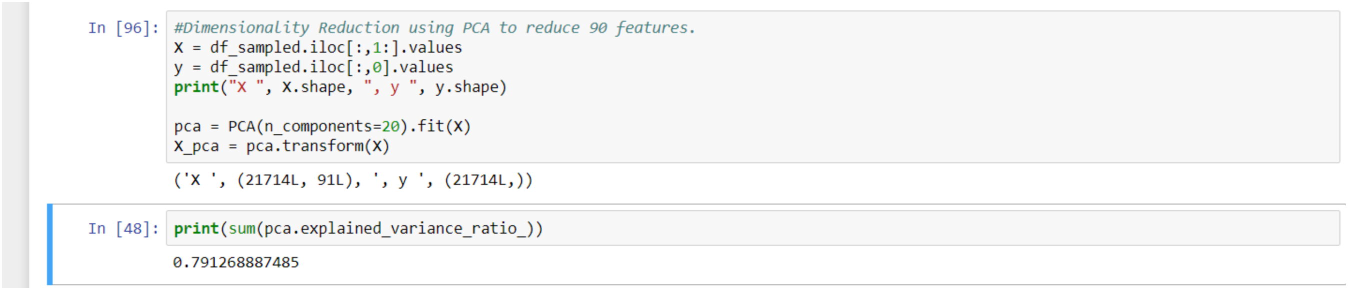
2.3 Baseline Model

Upon inspection of the figure above, we can say that the maximum number of songs like in the decade 2000. Hence, a good baseline would be to take 2000 as our baseline.

In our case, we have a baseline accuracy of 58%

2.3 Dimensionality Reduction

We tried performing PCA in order to reduce the number of features.



In [96]:
```
#Dimensionality Reduction using PCA to reduce 90 features.
X = df_sampled.iloc[:,1:].values
y = df_sampled.iloc[:,0].values
print("X ", X.shape, ", y ", y.shape)

pca = PCA(n_components=20).fit(X)
X_pca = pca.transform(X)
```
```
('X ', (21714L, 91L), ', y ', (21714L,))
```

In [48]:
```
print(sum(pca.explained_variance_ratio_))
```
```
0.791268887485
```

As seen from the screen shot above, the variance explained after performing PCA was 0.79. Since this value < 0.95, we decided to go ahead with performing our experiments on all the 90 features.

2.4 Experiment

2.4.1 Logistic Regression

$$\frac{F(x)}{1-F(x)} = e^{\beta_0+\beta_1 x}.$$

Logistic regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. We obtained an accuracy of 40.96% using this approach

2.4.2 Support Vector Machines

$$\vec{w} := \sum_j \alpha_j c_j \vec{d_j}, \quad \alpha_j \geq 0,$$

support vector machines (SVMs, also support vector networks[1]) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other.

We implemented the support vector machine algorithm in python sklearn package using a 3-fold cross validation on the dataset to predict the release year. The accuracy predicted was 58.01%

```
             precision    recall  f1-score   support

       1950       0.62      0.71      0.66       926
       1960       0.44      0.50      0.47       946
       1970       0.41      0.43      0.42       920
       1980       0.45      0.41      0.43       931
       1990       0.38      0.31      0.34       957
       2000       0.33      0.31      0.32       914
       2010       0.44      0.43      0.44       921

avg / total       0.44      0.44      0.44      6515
```

2.4.3 Random Forest

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

Upon implementation of the random forest algorithm with max Depth = 2, we were able to obtain an accuracy of 28.02%

3 Analysis

The Results of our experiments are tabulated below

Classifier	**Accuracy**
Baseline	58%
SVM	58%
Logistic Regression	40.9%
Random Forest	28.02%

From the Table above, we can infer that the SVM comes closest to the baseline. This is mainly due to the bias in the dataset itself. As was pointed out earlier, maximum number of songs in the dataset were released in the decade of 2000-2010. This gives us an overwhelmingly large accuracy for the baseline model.

3 Future Enhancements

- Include a config file where we can configure the number of peers, the peer names, port numbers and the directory paths before running setup.
- Extend this approach to multiple machines over a distributed environment
- Replace simple copy with the SFTP file transfers for downloads
- Implement a GUI