Predicting Song Hotness

Violet Dong
Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
xdong1@andrew.cmu.edu

Angus Fung
Department of Statistics and Data Science
Carnegie Mellon University
Pittsburgh, PA 15213
afung@andrew.cmu.edu

1 Introduction

The objective of this project is to explore the correlation between the hotness of a song and the features associated with it using the million song dataset. We proposed this problem because music is an integral part of our lives and there are songs more popular than the other. We would like to know what makes a song "hotter" than other ones, in particular which of the features provided by the million song dataset make the most difference. We would infer this information from the coefficients of the linear regression. This problem could be interesting because the model and the inference could provide information for songwriters to identify areas that they could work on to produce a successful song.

The million song dataset contains, as the name suggests, a million songs. The data for each song is stored in each individual h5 file. All the songs in the dataset are contemporary popular music tracks. In total, there are 54 features associated with each song. These features are either floats, integers or arrays. Of these 54 features, some of them are factual information such as song id, song name and year while others such as artist hotness and danceability are descriptive estimations calculated by algorithms. In the initial feature selection step, we only selected descriptive features that were either floats or integers. This is because factual information such as the album name is less likely to be independent of the success of the song. We would like to explore what it is about the music itself that makes a song stand out. However, it is worth noting that song id was a part of the initial feature selection because a unique identifier was needed to name each file. In addition, we discarded confidence values. By this step we were able to narrow down to 15 features. Once we accessed the data points on the jupyter notebook, we noticed that the dataset was rather incomplete. For example, the predictive variable, song hotness, had a large volume of null values, posing a challenge in the analysis. Although many other features did not contain missing values, they contained illegitimate values. For example, the variable danceability and energy were both measured between 0 and 1. However, a 0 in both of these variables actually meant that this particular feature was not analyzed, posing another challenge in the data processing stage. These features, along with analysis sample rate, were removed during cleaning of the data. The table below summarizes the final 10 explanatory features used to predice song hotness.

Table 1: Final Features Used

artist_familiarity	artist_hotttnesss	year	start_of_fade_out	tempo
duration	end_of_fade_in	key	loudness	mode

2 Methodology

The first step in the methodology pipeline was to store the data files on S3. As mentioned in the introduction, each individual song was wrapped within a h5 file, which was inconvenient to analyze. To solve this problem, we modified the conversion code provided by the course to convert the files from h5 to csv. In particular, we extracted information from each h5 file, transformed it into a row

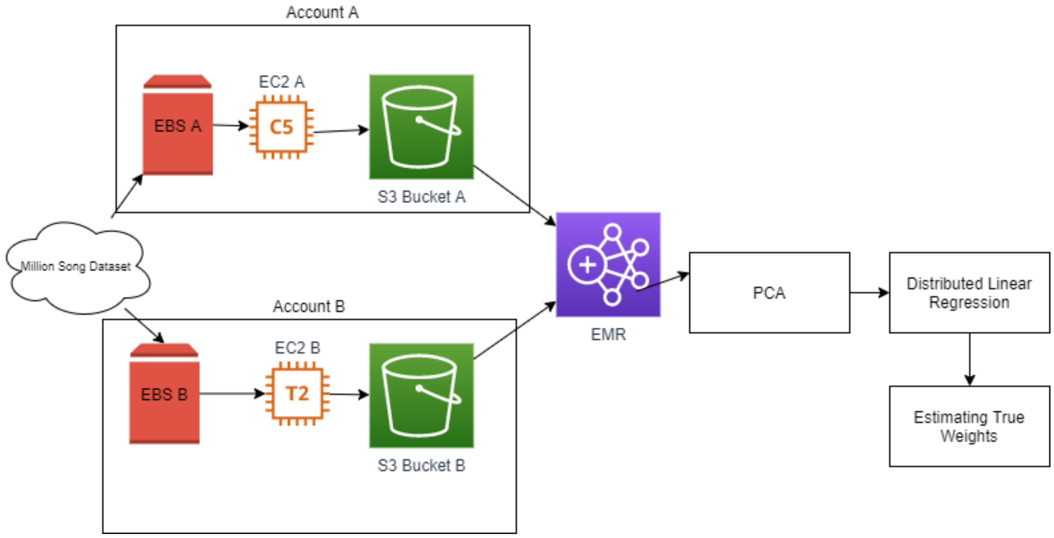

Figure 1: Machine Learning Pipeline

and combined 10,000 rows into a single csv file to upload to S3. The conversion process was split between the two partners. Since the dataset volume was divided among 26 folders, each team member processed 13 folders and uploaded the files to their S3 buckets. This conversion step also incorporated the initial feature selection since only 15 features were selected to be converted.

Once the files were uploaded to S3, we imported them from there to jupyter hub, in which we analyzed the data. As mentioned in the introduction, the predictive variable, song hotness contained many missing values. Since it was the predictive variable, it did not make sense to fill in the missing values with the feature average or random sampling. We decided to remove the rows where song hotness is null. In addition, we removed features that seemed to exhibit only value. The removed features included anceability, the analysis sample rate, and energy features

After cleaning the data, we performed principal component analysis in order to identify the most important components and thus try and reduce the dimensionality of the data while capturing important traits.

After performing dimesionality reduction, we regressed song hotness on the five most important principal components. We chose a linear regression model to model the relationship between the features and song hotness. Using MLlib from Spark, we attempted to fit a linear regression model using stochastic gradient descent, which convergences in less time at the cost of more communication. Another advantage of using linear regression was that it is a simple model that is easily interpretable, meaning that we can infer about the importance factors just from examining the weights.

Initial attempt at the stochastic gradient descent did not yield convergence. After trying multiple choices of k (top eigenvalues) ranging from 2 to 5, the weights of the regression continued to increase exponentially as the number of iterations increased. The weights calculated were thus nan. Therefore, we eventually decided to proceed with the closed form solution as it did not depend on convergence and could be done in a distributed fashion. Because we conducted dimensionality reduction, the dimension was small enough where a closed form solution would not take to long to compute.

After finding the linear regression model trained on the principal components, we used the principal components weights found to estimate the weights of the linear regression model trained on the original data. To do this we simply matrix multiplied the principal component matrix by the weights found from regressing on the principal components.

Using the estimated weights of the linear regression model trained on the original data, we assigned importance to features based on the magnitude of their coefficients. We believe that more important features would have larger coefficients, as a change in a feature with a larger coefficient would have a greater effect on the predicted than a feature with a coefficient close to zero.

3 Computation

The biggest computation issues centered around the sheer size of the data. Processing and transferring the raw data to S3 was the biggest computational challenge. To overcome this issue, we split the work between two accounts so that the raw data could be processed in parallel. Then when we processed the raw data, we selected only numerical features, which meant that we reduced the amount of data needed to be processed and sent to S3. These numerical features were also preselected, meaning that we didn't filter over all 54 original features, helping cut down on computation.

Another computation issue was convergence in linear regression. Originally we wanted to use gradient descent to find a linear regression model to predict hotness. Gradient descent seemed to be a good idea as it performs well on high-dimensional data, something we were concerned about early in the project. However in development we encountered issues with convergence, as when we reduced the number of iterations, we saw that the predicted weights were approaching infinity. Fortunately, we used PCA to find a reduced representation of the original data. Using this reduced representation meant that we reduced the dimensionality of the training data, meaning we could use the closed form solution to find the weights in a reasonable amount of time. Once we found these weights, we could reconstruct weights on the original data.

We used Amazon Web Services as our cloud service provider. We used two machines split between two team members to process and transfer the raw Million Song Dataset in parallel. One team member used a c5.xlarge instance because c5 instances are compute optimized and c5.xlarge is the largest available c5 instance for AWS Educate accounts. The other team member, however, used a t2.medium instance, which performed suboptimally. c5.xlarge was able to convert the files and upload them to S3 2.5 hours while t2.medium took 6 hours. We used three machines in our AWS EMR cluster, one master and two core machines, to analyze the data. Our master machine was a m5.xlarge machine, as it was recommended to be the default instance for master machines. Our core machines were also m5.xlarge also because it was the default instance used. We used Python exclusively, and made use of pyspark, specifically the sql module, and numpy. The overall cost of developing and computing costed about $30 between both members.

4 Results

After running our model on the dataset in a distributed fashion using the method described above, we were able to generate a list of coefficients indicating the significance of each feature. We inferred that the more impactful features would have coefficients with magnitudes larger than 0. By this criterion, we concluded that loudness, tempo as well duration were the three most important factors in a song's success.

Based on our analysis, we found that the longer a song is the more popular it is. One reason is that longer songs capture more dynamic and information. People are more intrigued by songs that do not simply repeat every phrase but continue for longer period of time. We suspect that people yearn for something beyond repetitive note.

We also concluded that tempo and loudness were key factors in a song's success. This conclusion makes more sense because the one million songs that we analyzed were contemporary pop music, which are faster and louder than songs from the last century just by observation.The rise of genres such as rap and EDM could be a contributor. Taking the US as an example, the top songs on all the music charts are often rap songs, which has a strong emphasis on tempo and rhythm.

Overall, our results make sense and are fitting to our society's standard today.