Predicting Preferences in Streaming Music
Ryan Kerr, Tony Li, George Zhang
CS181 Practical 3 Writeup
https://github.com/gzhang01/cs181practicals/tree/master/practical3

# 1 Abstract

Our task in this practical was to predict people's taste in music. That is, given data on users' ages, genders, and geographical regions, as well as a training set of users' play counts of artists, we sought to predict the number of times a given user will play a given artist. To accomplish this, we investigated unsupervised learning, using K-means to cluster the artists, as well as supervised learning, using a k-Nearest Neighbors approach. Neither of these approaches produced substantially effective predictions. We ultimately used modified user-median data to predict the number of plays given a user-artist pair.

# 2 Technical Approach

The naive approach that we start with as a baseline is as follows: for each user $u$, compute the median number of listens for the artists that user has listened to, $\text{MED}_u$. Then for a given user-artist prediction, simply predict $\text{MED}_u$. This is a reasonable first start because it takes into account the fact that some users listen to music more than others, but it fails to incorporate any data about the artists that we are predicting for. For example, using this method, a 13-year old girl from the US is just as likely to listen to Justin Bieber as she is a death-metal band from Finland. We tried two models to approach this deficiency.

**Exploring K-Means**
In order to incorporate information about the artist in our predictions we wanted to make distinctions between different kinds of artists. The most obvious kind of clusters would be by genre, but since that data was not available we decided to use an unsupervised clustering algorithm, namely K-Means, with the data we had.

Initially we created a very large data frame with a row for each artist and a column for each user, and the values corresponding to the number of listens between that artist-user pair. However, this resulted in a large, sparse data frame (most users did not listen to most artists), which made clustering very unsuccessful even after normalization. Therefore we eventually clustered the artists using the country and gender of the users. We ultimately decided to use 5 clusters (see Conclusion for rationale).

After doing this, for each artist $a$ we have a corresponding cluster numbered 1 through 5. For each user, instead of having one median for all artists, we then computed the median number of listens for cluster of artists, giving us five different medians $\text{MED}_{u,1} \ldots \text{MED}_{u,5}$. When predicting a given $(u, a)$ pair, instead of just predicting $\text{MED}_u$ we look at the corresponding artist's cluster $c$ and predict $\text{MED}_{u,c}$.

**Exploring k-Nearest Neighbors**
Another approach we used was k-Nearest Neighbors. The idea behind this approach is that users of the same gender, age, and location may tend to have similar music tastes and thus may provide better information with which to predict number of plays for a given artist. Thus the goal was to aggregate the data based on the artist, user's location, user's gender, and user's age.

We initially tried to use sklearn's kNN implementation, but quickly discovered that they require the input vectors to only contain numbers. This would require us to assign numerical values to countries, artists, and genders. Since it wasn't immediately obvious how to assign numbers to countries and artists and a simple assignment of 0 to one gender and 1 to another gender may not result in the same scale as the other variables, we decided not to proceed with this path. Instead, we wrote our own kNN algorithm.

We created a kNN class with fit and predict methods. Our approach to fitting went as follows: we created a dictionary and hashed first by artist, then country, then gender, and finally age. The reason for this structure was because we wanted as little repetition as possible. We thought artists would be the largest category and should be hashed first, and country would be the second largest and should be hashed second. While gender is a smaller category than age, we wanted to be able to select a range of ages if necessary, so we decided to hash that last. The dictionary stored lists of plays given that artist-user combination, which includes all of the training info. With this structure, we could then predict given an artist-user pair by indexing into the dictionary based on the artist and the user's information. We would then have all the play data for that artist and similar users, and we would generate our prediction by taking an unweighted average of the play values in the list.

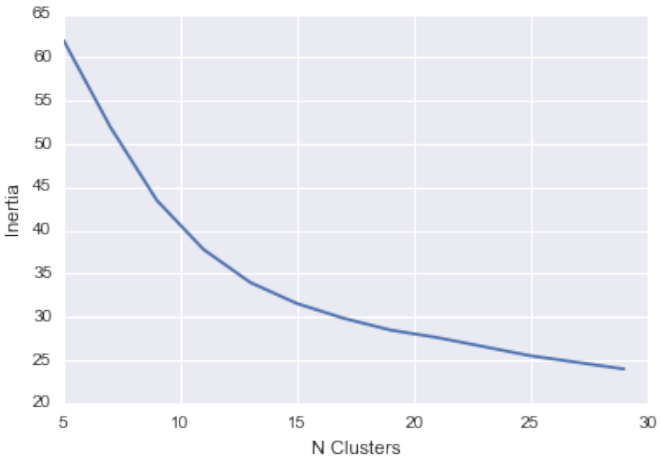
**Exploring Modified User-Median**
As we discuss in the Results section below, neither of the above two methods produced adequate predictions. As a result, we went back to the naive user-median method of predicting and looked harder at the data. We found that there were some user-medians that were relatively high – north of $10,000$. We reasoned that predicting such a large number for any given user-artist pair containing that user would be unlikely to give good predictions. Thus, if we assigned a large median to all of a user's artists, we would surely get a larger error. As a result, we decided to scale all user-medians above a certain threshold down, which would on average produce better results. We experimented with several values for a cutoff user-median and calculated a root mean square deviation from cross-validated training data to choose a relatively good cutoff.

# 3 Results

On the next page, we have a plot of the total error of the dataset as a function of number of clusters for our K-Means algorithm.

As we can see, there is not clear "elbow" at which we should limit the number of clusters. Traditionally if the number of clusters is not known beforehand, this kind of plot can provide guidance by selecting the number of clusters at which there is a distinct elbow in the graph. We ultimately decided on the 5 clusters for reasons described in the conclusion.

The modified median estimation using the clusters unfortunately gave worse results than the baseline. 80% of the predictions were equal to or worse than the baseline median prediction and only 20% were better. Thinking that this might be due to small sample size for a given user-cluster pair, we imposed a limit where we only used the cluster median if the number of observations for that cluster were greater than a certain threshold, but no threshold gave results that were better than the user median baseline.




Our kNN approach also did not produce great results. To test our kNN approach, we separated the training data into a train set (containing 90% of data points) and a test set (containing 10% of data points). We used the train set to train our kNN algorithm and then made predictions from the test set. Comparing the test set to the actual observations using the evaluation expression listed in the spec resulted in an MAE value of 263.18. This number is much higher than both benchmarks, so we did not submit it on Kaggle, and we decided this method was not worth pursuing further.

Disappointed by these results, we went back to the data and looked for useful patterns to enlighten our approach. This is when we noticed that some users had extraordinarily many plays of certain artists, and we reasoned that these outliers could affect our predictions adversely. Thus, we returned to our exploration of the median plays for each user, and we decided one solution would be to cap the median number of plays a user could have. This would mitigate the effects of the audiophiles by not skewing our predictions so much for these users. This solution intuitively seemed useful, because we reasoned these audiophiles were more likely to listen to a handful of artists devoutly but not as much to most other artists. Thus, we did not want to assign a large median extremized by a user's zealous taste in niche music to all the artists in the dataset.

We experimented with different values for the user-median cutoff. Lower values appeared to increase our calculated deviation in our cross-validation, and values much higher than 10,000 didn't have much of an effect on the deviation of our predictions, as few users had such large medians. Thus, we settled on a cutoff of 10,000 plays and submitted the results to Kaggle for a public leaderboard score of 137.78858, which was just lower than the benchmark.

# 4 Conclusion

We pursued both the K-Means and kNN approaches because we wanted to incorporate information about both the artist and the user's individual preference for similar artists. In both approaches we effectively narrowed down the scope of the data that we based our predictions on. In the K-Means approach, a user usually had about 1/5 of his or her total artists in any given cluster. This significantly reduced the meaning of the median value, and created more noise than in the overall median baseline prediction. With the kNN approach, a number of the artist-country-gender-age groupings had relatively few observations, which introduced the same problem. Even more significantly, there were certain test data points whose groupings were simply not in the training data. Unfortunately, this meant that we had to find "close" data points, which was slightly arbitrary when we were missing certain combinations such as artist-country pairs.

This problem with sample size is important to note. By implementing these methods the way we did, we effectively threw away a large proportion of a user's data for any given prediction. Even though these models accounted for nuances in the differences between artists, that was not enough to make up for the ignored data.

This problem with sample size is part of the reason why we chose 5 clusters. With a larger number of clusters, the number of samples for a given user-cluster pair will become smaller, thus exaggerating the sample size problem we have been running into. We found that 5 clusters were reasonable enough to capture variation between artists but were not too large so that the data was unusable (although, it seems like even this small number of clusters was not enough to address that problem!).

Another weakness of our kNN approach was that it did not incorporate information about the individual user. In the user median baseline, the prediction for a given user-artist pair was entirely based on the user's past listening tendencies. With our kNN approach, the prediction is entirely based on the tendencies of other similar users. This is valuable information, to be sure, but it glances over what may be the best individual predictor.

Ideally we would be able to incorporate the user-artist interaction without ignoring parts of the user's individual data. If we had more time we would have explored ways to incorporate this data in the form of some kind of multiplier, or matrix factorization as used in the research papers, as opposed to a filter. The techniques we used were filtering techniques in that they filtered down the data we looked at to get a more specified view.