Assignment 2 Writeup

Yousef Hawas and Anshul Shah

Dataset

Our dataset comes from Last.fm, an older music website from 2002 in the United Kingdom that allows users to listen to thousands of artists, become friends with other users, and add descriptive tags to an artist for other users to see. A link to download the dataset and a data dictionary can be found here: https://grouplens.org/datasets/hetrec-2011/

1. Literature Review

Data from Last.fm is commonly used among research papers exploring approaches to making recommendations. This dataset was actually released during the "2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems" in 2011. The conference intends to shed light on approaches to recommender systems using multiple sources of contextual information, such as tags, listener count, reviews, etc.

Many of the papers in the conference that year focused on using features such as popularity and temporal features of songs to create user recommendations. There was even a workshop focused on discussing the best evaluation method and raised a point about judging success from the user's point of view (whether the user enjoyed the new product or song recommended) [1].

A few papers in this conference also focused on making accurate recommendations and predictions in the "long-tail," which includes the less popular artists or users where less data is included [2]. There tends to be a bias towards predicting the most popular artists, and the performance accuracy for predicting less popular items is usually low, prompting attempts at using extra information to address this bias. For example, a paper that uses the Last.fm dataset created a prototype recommender system specifically for predicting less popular artists [3].

Another specific analysis of the Last.fm data uses "implicit user feedback" in the form of number of listens to make recommendations [4]. The author uses matrix factorization and assigns estimated ratings between each user and artist based on the number of listens.

We apply some general strategies used by some of the authors we've cited, such as leveraging tags and friendship information to make predictions as well as testing the usefulness of popularity as a feature in our model.

2. Descriptive Analysis

Some high level summary statistics were provided in the downloads material, replicated in the table below

Category	Value
Users	1892 users
Artists	17632 artists
User-friend relations	12717 relations

Average friends per user	13.4 friends
Average artists listened by user	49 artists
Average users that listen to each artist	5.27 users
Average number of listens by each user	36566 listens

Furthermore, in order to accomplish our predictive task, which will be discussed at length later in the report, we performed negative sampling on the data. This helped us perform some supplementary analysis and provided us with insight on the following aspects of the interaction data.

Popularity

Although the data gives us information on how many times each user has listened to an artist, we choose to ignore this feature when defining popularity because a single user or a small group of users could artificially inflate an artist's popularity by being extreme outliers in how much they listen to an artist. When ranking artists by popularity and deciding which artists to include as a “popular” artist, we use the number of listeners, which aligns more with the conventional meaning of popularity when talking about artists.

The most popular artists in terms of the most number of users are listed below:

1. Lady Gaga (611 users)
2. Britney Spears (512 users)
3. Rihanna (484 users)
4. The Beatles (480 users)
5. Katy Perry (473 users)

These are extreme outliers from most of the artists represented in the data as the average artist in the dataset has just over 5 users. However, since there are only about 1900 users in the data, the most popular artists capture a large proportion of users. In fact, 71 percent of the users have listened to at least one of the top 10 artists and 59 percent of users have listened to at least one of the top 5 artists.

After generating negative samples, we are able to get separate statistics when a user-artist relation is TRUE (i.e. the user is a listener to the artist) as opposed to when a user-artist relation is FALSE (i.e. the user is not actually a listener to the artist).

When a user-artist relationship is FALSE, the average number of users that listen to that artist is only around 4 listeners, whereas the number of listeners for an artist in a TRUE user-artist pair is about 68.7 listeners. This indicates that popularity could be a useful indicator in predicting whether a not a user listens to an artist.

Friendship

Last.fm allows users to “friend” each other and see which artists their friends listen to. Although the average user has 13.4 friends, the distribution of the number of friends per user is shown below in Figure 1.

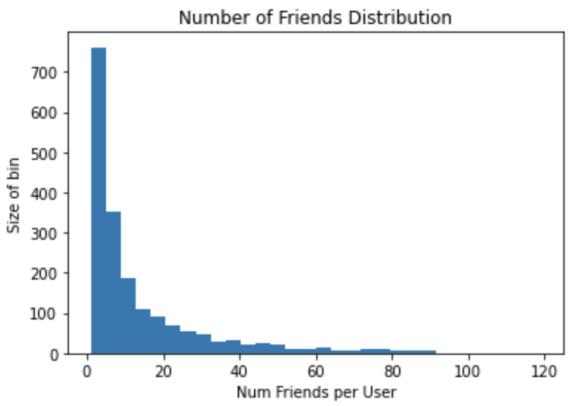

Figure 1

A large proportion of users have only 1 or 2 friends, whereas users with the most number of friends (over 100) are clear outliers.

We are able to run a similar analysis to the popularity analysis where we compare the group where a user-artist relation is TRUE to the FALSE group.

Within the negative samples where the user does not listen to the artist, the average number of friends of that user who did listen to the artist is only 0.026. However, among the samples where the user does listen to the artist, the average number of that user's friends who also listen to the artist is 1.9 friends. The stark difference between the two values indicates that the number of *friends* that listen to an artist could be a useful predictor of whether or not a *user* listens to an artist.

Artist Tags

Another interesting feature of our dataset is that users can generate "tags" to assign to an artist. Each artist can have multiple tags (most artists do) and each tag can be applied to multiple artists (most tags are). The five most popular tags present in the data are:

1. "rock" (2283 artists)
2. "electronic" (1749 artists)
3. "alternative" (1743 artists)
4. "pop" (1739 artists)
5. "indie" (1537 artists)

Again, by running an analysis of the negative samples vs. positive samples, we compared the average maximum similarity between an artist of interest and other artists the user listens to. When we look at the FALSE artist-user relationships, the average max similarity with other artists is 0.056 compared to an average max similarity of 0.225 for TRUE relationships. This provides evidence to tag similarity being helpful for predicting whether a user would listen to an artist.

3. Predictive Task

For our predictive task, we opted to go for a binary classification task. For every (user, artist) interaction we decided to classify whether or not that user would listen to the artist. In order to effectively tackle this task we performed negative sampling on the set of (user, artist) interactions. For every positive interaction, we randomly generated a negative interaction consisting of that user and an artist they have not been observed listening to.

Evaluation Criteria

In order to evaluate the performance of our models, We split our (user,artist) interaction set into a training set and test set using an 80/20 split. Then the model's performance was assessed using the accuracy on the test set which is defined as:

$$Accuracy = \frac{\#\ of\ correct\ predictions}{\#\ of\ datapoints}$$

We compared our experiments vs a baseline heuristic model as well as various models that utilize different combinations of our feature set.

Feature Set

During our exploration of the dataset, we identified some potential predictors to use in our classification and they are defined as follows:

- *Popularity*: number of unique users that have listened to a given artist
- *Max User Jaccard*: Given a (user, artist) pair, it is defined as the max jaccard similarity between the artists the user has listened to and the given artist.
- *Max Tag Jaccard*: Given a (user, artist) pair, it is defined as the max jaccard similarity between the tag set for that artist, and the tag sets for the artists the user has listened to.
- *Friend Count*: Given a (user, artist) pair, it is defined as the count of the user's friends that have listened to that artist.

In order to obtain these features, some data structure manipulation was needed. All these features were obtained using dictionaries to map different things. For popularity, we had to map artists to the set of users that have listened to them. For the similarity based features, we needed to map users to their listened artists and map artists to the tags that were assigned to them. Finally, for the friend count we had to map users to their friends as well as use previously defined dictionaries that map users to their listened artists.

Main Baseline Model

Our main baseline model used for comparison is a simple heuristic function based on some of the aforementioned features. The predicted label is defined as follows:

$$\begin{aligned} Label = &(Popularity > \lambda_1) \wedge \\ &(MaxUserJaccard > \lambda_2) \wedge \\ &(FriendCount > \lambda_3) \end{aligned}$$

These thresholds were set experimentally, and the best performance obtained by this baseline was an accuracy of ***79%***, which is a fair baseline to compare against.

4. Proposed Model

For our model, we tested the hypothesis that an ensemble of our features will outperform our baseline. Therefore we experimented with combinations of our features fed into various classifiers. We built a baseline heuristic function based on our features and compared it's performance with our proposed model that utilizes python's scikit-learn library's *LogisticRegressor*. The idea behind this is that an ensemble method on our features will be able to outperform our baseline as the regressor would be able to provide more complexity than a simple heuristic method.

Optimizing Performance

In order to optimize the model's performance on the test set. We experimented with different combinations of our features in order to get a better idea of what features perform well for this task on the dataset.

Hyperparameter Tuning

In order to obtain a better generalized model, we had to tune the LogisticRegressor's hyperparameters. We combatted overfitting by setting a stronger regularization strength (C parameter) and evaluated the training accuracy as well as the testing accuracy in order to obtain a better balance. Furthermore, we decreased the number of iterations performed by the model during training so as to prevent the model from becoming too complex and overfitting. Last but not least, we used a TRUE value for our fit_intercept parameter as our features are mostly indicators of a positive label (as they range from 0 to some positive value).

Other Attempted Models

We also considered using SVC (support vector machine classifier) as an alternative option to our Logistic Regressor. The idea behind this was that it offers a different approach to classification as it tries to optimize the number of mistakes made by the model as opposed to the logistic regressor's approach. However, we found that it tended to perform quite similarly to our Logistic Regressor. Furthermore, the SVC algorithm scales very poorly to the number of samples in the training set, and takes a long time to train.

Heuristic functions were also used in an attempt to create a simple but powerful model, but they were consistently outperformed by our ensemble classifiers.

Strengths and Weaknesses

Through our experimentation, we have identified some strengths and weaknesses to each of our approaches.

For heuristic models, they are quite simple to implement and don't require much hyperparameter tuning besides setting your thresholds. This means that these types of models are much less prone to overfitting on training data. However, this comes at the cost of the model's complexity and performance.

For logistic regressors, they are simple to implement and can often provide impressive performance when feature engineering is done well. Furthermore, the sigmoid property allows us to get predictions that are bounded between 0 and 1, allowing us to represent things in a probabilistic manner. The parameters of the model are also highly interpretable and can give insight on the importance of our features.

As for SVCs, they tend to fit better on non-linear data. However, the algorithm takes too long to train when the size of the training set is relatively large.

5. Results and Conclusions

Through our experiments with different feature sets we were able to obtain the following results using logistic regression:

Model Features	**Test Accuracy**
Popularity + Max User Jaccard + Friend Count	81.4%
Popularity + Max User Jaccard + Max Tag Jaccard	84.3%
Popularity + Max User Jaccard + Max Tag Jaccard + Friend Count	84.5%

Max User Jaccard + Max Tag Jaccard + Friend Count	85%
Max User Jaccard + Max Tag Jaccard	85.8%

We were able to conclude that our ensemble method combined with our choices of feature set significantly outperformed our heuristic-based baseline model (79% accuracy). The significance of this is that we now have a model that can predict whether or not a given user in the dataset will listen to a certain artist, with an accuracy of approximately 86%. This can give insight on whether or not to recommend a certain artist to a user or not.

Feature Performance Conclusions

Our different combinations of features gave us some insight on which features work for this task on the dataset, and the conclusions are as follows:

Popularity- ineffective

- Popularity appears to not be an effective predictor in this case. Possible reasons for this could be caused by outlier artists (artists with an abnormally large amount of listeners or artists with an abnormally small amount of listeners. Another reason could be that popularity as a feature makes the assumption that a popular artist will be listened to by most users, which is not always the case.

Friend Count- ineffective

- Much like the popularity feature, this feature was not the best predictor. Perhaps this is because it implies that friendships between users are based on a shared musical taste, which may not be the case. Users could be friends based on other factors as well.

Max Tag Similarity- effective

- A user is more likely to listen to an artist if that artist has relatively similar tags to other artists the user has listened to. Logically, this makes sense since many users are fans of genres, so they would listen to similar artists in that genre.

Max User Similarity- effective

- A user is more likely to listen to an artist if other users with a shared taste have listened to the artist. This follows from the same logic in the previous feature, since a user probably listens to the same artists that similar users listen to.

Model Interpretation

The parameters we obtain through training can be interpreted and some meaningful conclusions can be drawn through their values. For our best performing model (a logistic regressor with similarities as features) our decision function is defined as follows:

$$Y = \theta_0 + \theta_1 \times (Max\ Tag\ Jaccard) + \theta_2 \times (Max\ User\ Jaccard)$$

θ_0: This parameter can be interpreted as our model's default prediction for similarities of 0. In our case, this value is negative because our features are all on the positive scale and therefore contribute towards positive predictions. In our best performing model this was equal to -2.9.

θ_1: This is the coefficient for the highest Jaccard similarity among other artists the user has listened to (based on artist's tag

sets). Our value is positive, meaning that the greater the similarity between the artist to predict and other artists a user listens to, then the more likely a user is to listen to that artist. In our best performing model this was equal to 9.9.

θ_2: This is the coefficient for the highest Jaccard similarity between other a user and other users who listen to the same artist. Again, this value is positive indicating that a greater similarity to other listeners means a greater chance that the user listens to the artist. In our best performing model this was equal to 16.5.

This equation defines a decision boundary for our task, where data points achieving a positive score will be classified as TRUE while ones with a negative score will be classified as FALSE.

Looking at the coefficients, we can also see that the *Max User Jaccard* is deemed to be more important towards prediction than the *Max Tag Jaccard*. This is due to the fact that both features lie in the same range of values, but the *Max User Jaccard* has a coefficient of higher magnitude.

6. Citations

1. Pearl Pu, Li Chen, and Rong Hu. 2011. A user-centric evaluation framework for recommender systems. *In Proceedings of the fifth ACM conference on Recommender systems (RecSys '11). Association for Computing Machinery, New York, NY, USA, 157–164.* DOI:https://doi.org/10.1145/2043932.2043962

2. Neel Sundaresan. 2011. Recommender systems at the long tail. In Proceedings of the fifth ACM conference on Recommender systems (RecSys '11). Association for Computing Machinery, New York, NY, USA, 1–6. DOI:https://doi.org/10.1145/2043932.2043934

3. Last, M., House, K., & Bosteels, K. (2010). Music Recommendation and the Long Tail.

4. Pacula, M. (2010). A Matrix Factorization Algorithm for Music Recommendation using Implicit User Feedback.