# Predicting Audio Features with Last.fm Tags

Jaime Ramírez Castillo[1*] and M. Julia Flores[1†]

[1]Departamento de Sistemas Informáticos, Universidad de Castilla-La Mancha, Campus universitario s/n, Albacete, 02071, Spain.

*Corresponding author(s). E-mail(s): Jaime.Ramirez@alu.uclm.es;
Contributing authors: Julia.Flores@uclm.es;
†These authors contributed equally to this work.

**Abstract**

In this paper, we discuss a number of experiments to analyze the suitability of music label representations to predict certain audio features, such as danceability, loudness, or acousticness ...

**Keywords:** Music information retrieval, Artificial intelligence

## 1 Introduction

Music information retrieval (MIR) is an interdisciplinary research field that encompasses the extraction, processing, and knowledge discovery of information contained in music. MIR research intersects with other fields, such as computer science, signal processing, musicology, and sociology. The field covers applications in recommendation systems, music classification, music source separation, and music generation, among others [1].

MIR applications often attempt to extract information from the music audio signal, but analyzing associated metadata is also a common practice. Audio signals are typically preprocessed and transformed into intermediate formats, such as frequency-based representations. Associated metadata, such as editorial information, lyrics, or user-generated content is usually in text format. However, metadata can be available as images of videos, for example, when analyzing album artwork, or music videos.

Depending on the specific MIR application, researchers or practitioners expect different output values. For example, an application that extracts audio features might return values for the tempo, the key, or the sample rate. In more complex applications, where, for example, machine learning is used, applications might return the estimated emotion that a track produces on a listener, or the predicted music genre of a particular track.

Many of the challenges in this field use audio features to predict other music-related aspects. Examples of these features are For example, some studies have been using these features to predict the probably of a track being a hit [REF].

To make these predictions, researchers have been using machine learning models, specially over the past few years. Previously, researchers commonly used hand-crafted audio computational models to perform these tasks.

But what if we tried to use these audio features as the predicted values? We would need a set of predictor variables to acurrately predict the value of audio features. This premise is the core concept of this article.

The idea is, given a set of tags, to predict a set of audio features that a hypothetic track would exhibit. Then, we could build a track selection algorithm that selects actual tracks that are the closest to the predicted audio features. This process could be part of an explainable recommendation pipeline, where users enter a set of tags, and they recieve the predicted audio features, the closest tracks to those features, and the distance values between each track and the predicted features.

In the remainder of the article, we explain the data gathering and preparation process, as well as the data input formats and varios models. We will explore various models for the same track and provide insights on how accurately the prediction can be, by using only Last.fm tags.

## 2 State of the Art

In recent years, researchers have used Last.fm tags in classification and regression tasks. Several studies have used Last.fm to predict music sentiment or mood.

In the last decade, Last.fm tags have been a popular source of metadata for MIR tasks. Last.fm tags can contain information the genre, mood, and style of music, and might be use to characterize certain features of a music piece.

Last.fm tags can be useful when the audio signal available, for example, due to copyright limitations.

A number of studies have explored the use of Last.fm tags in MIR, and have shown promising results in predicting various audio features.

Researchers have used Last.fm tags. For example, Laurier et al. analyzed how Last.fm tags categorize mood. In their study, they created a semantic mood space based on Last.fm tags [2].

For example, Çano and Morisio discuss the process they follow to create a dataset of music lyrics annotated with Last.fm. In the creation process, they conclude that Last.fm tags are mostly related to music genre and positive moods [3]. In a similar direction, Bodó and Szilágyi generated a dataset for lyrics genre classification by combining the Last.fm with MusicBrainz data [4]. MusicBrainz [1] is an online database of music editorial metadata. [1]

The Last.fm data has been the most widely used Last.fm dataset[2] in research. This dataset is a complementary dataset of the Million Song Dataset (MSD) [5].

Additionally, the Spotify audio features have been used in multiple studies. Historically, these features were also called "EchoNest audio features". Echonest was an online platform was was later acquired by Spotify.

Wang and Horvát use audio features to study differences between male and female artists [6]

In general, these studies confirm the possibility of extracting knowledge from Last.fm tags. To the best of our knowledge, no studies have addressed the problem of audio features regression, based solely on Last.fm tags.

Jamdar et al. used EchoNest audio features, combined with lyrics data to classify songs into emotion tags. These classes were first defined based on a Last.fm tags emotion mapping [7].

Similarly, Non-negative Matrix Factorization was applied in combination with EchoNest audio features for song recommendations [8].

P4kxspotify is a publicly available dataset that combines music review texts with Spotify audio features. The dataset creators argue that, although the terms of service prohibits scraping, their work is ethical [9].

In general, Spotify audio features have been used as predictive variables. We, to the best of our knowledge, are unaware of students that uses these features as target variables.

### 2.0.1 Machine Learning Models

Several studies have explored the use of machine learning models to predict audio features from audio metadata, such as Last.fm tags.

The experiments conduced in this study used two classical regression models, Boosted tree regressor and Bayesian ridge regressor. These two machine learning models that have been widely used in MIR.

Boosted tree regressor is a decision tree-based model that sequentially adds weak learners to the model to improve its performance. Bayesian ridge regressor, on the other hand, is a probabilistic model that uses Bayesian inference to estimate the parameters of the model.

Additionally, language models, such as GTP-2 have shown promising results in various natural language processing (NLP) and generation.

In this particular study, GTP-2 has been fine-tuned for regression tasks, and has shown good performance in predicting audio features from metadata.

[1] https://musicbrainz.org/

[2] Last.fm dataset, the official song tags and song similarity collection for the Million Song Dataset, available at: http://millionsongdataset.com/lastfm.

This paper aims to apply the boosted tree regressor, Bayesian ridge regressor, and GTP-2 models to predict Spotify audio features from Last.fm tags. The experiments compare the performance of these models and evaluate their effectiveness in predicting various audio features. The results of these experiments will provide insights into the use of different machine learning models for MIR tasks and can have practical applications in music recommendation systems and genre or mood recognition.

To be Continued . . .

# 3 Generating a Dataset

Before conducting experiments on predicting audio features from tags, we constructed a dataset, by gathering the data from the Last.fm and Spotify APIs.

## 3.1 A Single-user Dataset

Similar to other intelligent systems, recommender systems must be trained, by using user preference data, to produce adequate recommendations. For our recommendation framework, we have leveraged the knowledge discovery potential of large historical listening logs, gathered from Last.fm.

To characterize the preferences and context of the user, we have chosen to start with a simple scenario, where just data from a single user is available. By training our system with data from a single user, we also want to begin a discussion, given the following question: Is it possible to train recommender systems, and in particular, user-centric systems, by using a single-user dataset?

To the best of our knowledge, research on user-centric rec- ommender systems has concentrated its efforts on explain- able AI Wang et al. [2019], and also user-centered evaluation of these systems Knijnenburg et al. [2012]. We also argue that recommender systems that exploit the preferences of a single user, or a reduced number of users, might as well be considered as user-centric models.

## 3.2 Last.fm

Last.fm is a online music service for uses to keep track of their music listenting habits. Last.fm can also be considered as an community where users tag artists, albums, and tracks, according the the own perception of the user.

For nearly two decades, users have been contributing to Last.fm by tagging tracks with arbitrary text labels. These tags do not necessarily have to be single-worded. Users often use short sentences to define a song, such as ‘I like this track‘, or ‘on the beach‘.

Community-contributed tags from the Last.fm API. These tags are text labels that Last.fm users assign to artists, albums, or tracks. Users apply these tags to categorize music from their own perspective, which means that tags do not fit into any structured ontology or data model. Tags can refer to aspects such as genre, emotion, or user context.

### 3.2.1 Last.fm Tags

Last.fm uses the term *scrobble* to refer to a single track playback, in a particular moment. We have queried the Last.fm API to download the user's scrobble logs, reported from 2007 to 2022. For each scrobble, we have gathered the following information:

- Track playback timestamp.
- Track MusicBrainz Identifier (MBID), if exists.
- Track name
- Artist name
- Track tags. If the track does not have any tags assigned, then artist tags have been used.

For each tag assigned to a track, or an artist, Last.fm includes a count property to indicate the popularity of the given tag for the track. Last.fm normalizes this value in the 0-100 range, so the most popular tag for a track can have a count value of 100.

Users normally listens to their favorite tracks many times, so the amount of individual tracks listened is much smaller than the number of track plays. In this case, the amount of individual tracks listened is about 20,000.

The format is as follows:

```
{
      "artist - name": {
            "eletronica": 100,
            "rock": 80,
            "pop": 45,
            "jazz": 0,
            "nu-jazz": 0,
            "country": 0,
            "soul": 0,
      }
}
```

## 3.3 Spotify

After gathering Last.fm data and identifying the unique tracks that represent the user music collection, we have collected Spotify audio features. For each of these individ- ual tracks, we have downloaded the Spotify audio features specific to the given track.

The Spotify audio features are numerical values that repre- sent high-level audio information computed from a specific track. These values characterize a track, musically speaking, by measuring relevant musical aspects.

The features provided by the Spotify API are acousticness, danceability, duration_ms, energy, instrumentalness, key, liveness, loudness, mode, speechiness, tempo, and valence. Table 1 describes these features. The reader can find further details about each feature in the Spotify API documentation 4.

A small portion of the tracks do not have features available in Spotify, so they have been filtered out from our experiments.

After filtering songs that miss Last.fm tags or Spotify audio features, our dataset contains 14009 samples.

*Track audio features from the Spotify API.* These are attributes computed from the audio themselves. They are a way to describe music by using numerical values. For example, a danceability attribute of 0.95 means that a particular song is highly suitable for dancing.

## 3.4 Last.fm Tags Representations for Training

### 3.4.1 Tabular

Each tag is a column and each cell contains the popularity value of a tag for a track. A cell is 0 if a tag is missing for a track.

The number of columns is limited to the top-K tags.

### 3.4.2 Tabular Tokens

Tags are converted to text tokens. Columns represent token positions, and cells contain the token at a particular position, for a track. To tokenize tags, we have used the GTP2 tokenizer. Because the tokenizer requires a string as input, we have converted the set of tags for each track into a string. To *stringify* the tags, we have concatenated tags with multiple strategies:

- By including tag popularity: 'rock 2, pop 1'.
- By repeating tags based on popularity: 'rock rock, pop'.
- By ordering by popularity: 'rock, pop'.

## 3.5 Training Data By Track

When generating training data by track, the tabular formats present sparsity problems.

For tabular representations, we need to defined a fixed set of columns as tags. For most of tracks, most columns are '0'.

The sparsity of a matrix is the number of zero-valued elements divided by the total number of elements (e.g., m * n for an m * n matrix) is called the sparsity of the matrix

The mean and standard deviation for each variable are as follows:

Figure 1 represents the density function of each variable.

We can compare our single-user data with a bigger audio features dataset https://www.kaggle.com/datasets/tomigelo/spotify-audio-features. This dataset shows similar data distributions.

**Table 1** Audio features description

| Feature | $\mu$ | $\sigma$ |
|---|---|---|
| Danceability | 0.599 | 0.193 |
| Energy | 0.631 | 0.233 |
| Acousticness | 0.221 | 0.302 |
| Instrumentalness | 0.514 | 0.382 |
| Valence | 0.435 | 0.279 |

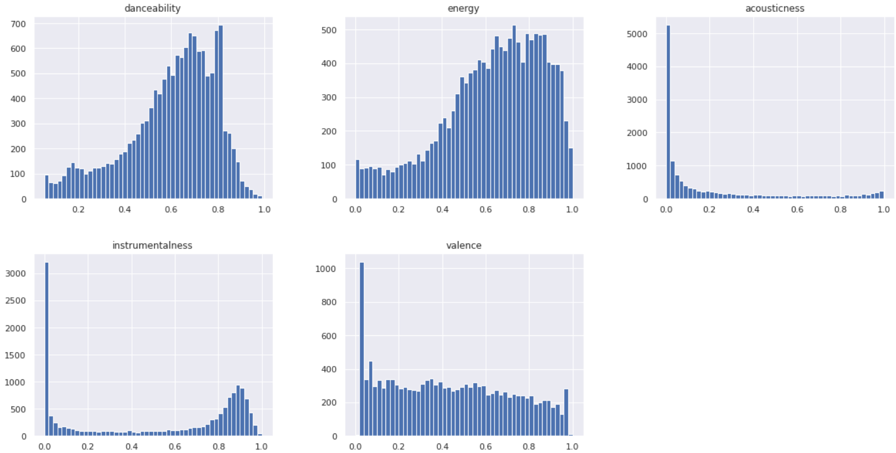

**Fig. 1** Distribution of audio features

I have posted a question on the Spotify Developer forum, relative to the release of a dataset https://community.spotify.com/t5/Spotify-for-Developers/May-I-publish-an-open-source-dataset-that-contains-Spotify-Audio/td-p/5529259

## 3.6 Formatting Input Data for Predicting From Last.fm Tags

The input data passed to the XGBoost regressor is formatted in tabular format, as follows:

- Given that $Tags_k$ is the set of most $k$ frequent Last.fm tags in the user listening history and, where each $tag \in Tags_k$.
- Given that $Audio$ is the set of Spotify audio features, where each $feat \in Audio$.
- For each $track$:
  - $X_{track,tag}$ is the strengh of $tag$ for $track$. This value is in the $0-100$ range.
  - $y_{track,feature}$ is the value of the audio feature $y$ for $track$.

An example of this data format is provided in table 2.

**Table 2** Tabular data format for Last.fm tags in XGBoost and Bayesian regressors

| Track | $X_{electronic}$ | $X_{ambient}$ | $X_{...}$ | $y_{energy}$ | $y_{valence}$ | $y_{...}$ |
|---|---|---|---|---|---|---|
| Massive Attack - Blue Lines | 62 | 6 | ... | 0.496 | 0.947 | ... |
| The Beta Band - Squares | 40 | 3 | ... | 0.446 | 0.507 | ... |
| ... | ... | ... | ... | ... | ... | ... |

## 3.7 Formatting Input Data for Predicting From Tokens

In this particular case, the $X$ values of the tabular input data are tokens. These tokens are obtained from passing the a string of concatenated Last.fm tags through a tokenizer. The formal definition of this data format is as follows:

- Given that $X_l$ is the token vocabulary, where $l$ is the maximum vocabulary length.
- Given that $Audio$ is the set of Spotify audio features, where each $feat \in Audio$.
- For each $track$:
  - $X_{track,n}$ is token found at position $n$, after tokenizing the tags string.
  - $y_{track,feature}$ is the value of the audio feature $y$ for $track$.

An example of this data format is provided in table 3.

**Table 3** Tabular data format for tokens in XGBoost and Bayesian regressors

| Track | $X_0$ | $X_1$ | $X_2$ | $X_{...}$ | $y_{energy}$ | $y_{valence}$ | $y_{...}$ |
|---|---|---|---|---|---|---|---|
| Massive Attack - Blue Lines | 101 | 5099 | 6154 | ... | 0.496 | 0.947 | ... |
| The Beta Band - Squares | 101 | 4522 | 2600 | ... | 0.446 | 0.507 | ... |
| ... | ... | ... | ... | ... | ... | ... | ... |

## 3.8 Formatting Input Data for Predicting From Text

When using transformer models, the input data is a string. We must represent the Last.fm tags, which are initially in the $(tagname, tagpopularity)$ form, to a a string.

After converting to a string, the formal definition of the input data is as follows:

- Given that $X$ is tags represented as text.
- Given that $Audio$ is the set of Spotify audio features, where each $feat \in Audio$.
- For each $track$:
  - $X_{track,n}$ is set of tags for $track$, encoded as a single string.
  - $y_{track,feature}$ is the value of the audio feature $y$ for $track$.

An example of this data format is provided in table 4.

**Table 4** Text data format for tokens in XGBoost and Bayesian regressors

| Track | $X$ | $y_{energy}$ | $y_{valence}$ | $y_{...}$ |
|---|---|---|---|---|
| Massive Attack - Blue Lines | "hip hop, chill, bristol, ..." | 0.496 | 0.947 | ... |
| The Beta Band - Squares | "alternative rock, folk, ..." | 0.446 | 0.507 | ... |
| ... | ... | ... | ... | ... |

# 4 Experiments

We trained a commonly used machine learning models to predict an audio feature, given the set of tags for a particular track.

- Boosted tree regressor [10]
- Naive Bayes Regressor [11]
- Fine-tuned GPT-2 model

## 4.1 Boosted Tree Regressor

We configured the boosted tree regressor model with the training parameters listed in table 5.

## 4.2 Naive Bayes Regressor

The Naive Bayes Regressor, and in particular, Bayesian Ridge, is the model used for regression in this case.

The training parameters are listed in table 6.

## 4.3 Fine-tuned Transformer

TODO

## 4.4 Experiments Execution and Results

The experiments:

### 4.4.1 Results for Tabular Data Models

# 5 Conclusions

TODO

# 6 Acknowledgments

TODO

**Table 5** Training parameters for XGBoost regressor

| Parameter | Value |
|---|---|
| objective | reg:squarederror |
| base score | 0.5 |
| booster | gbtree |
| colsample bylevel | 1 |
| colsample bynode | 1 |
| colsample bytree | 1 |
| gamma[1] | 0 |
| learning rate | 0.300000012 |
| max delta step | 0 |
| max depth | 6 |
| min child weight | 1 |
| estimators | 200 |
| n jobs | 12 |
| num parallel tree | 1 |
| predictor | auto |
| random state | 0 |
| reg alpha | 0 |
| reg lambda | 1 |
| scale pos weight | 1 |
| subsample | 2 |
| tree method | auto |

[1] Minimum loss reduction required to make a further partition on a leaf node of the tree.

**Table 6** Training parameters for XGBoost regressor

| Parameter | Value |
|---|---|
| Maximum iterations | 300 |
| Tolerance[1] | $1 \times 10^{-3}$ |
| alpha 1 | $1 \times 10^{-6}$ |
| alpha 2 | $1 \times 10^{-6}$ |
| lambda 1 | $1 \times 10^{-6}$ |
| lambda 2 | $1 \times 10^{-6}$ |

[1] Tolerance for the stopping criteria.

# References

[1] Ramirez, J., Flores, M.J.: Machine learning for music genre: multifaceted review and experimentation with audioset. Journal of Intelligent Information Systems **55**(3), 469–499 (2020)

[2] Laurier, C., Sordo, M., Serra, J., Herrera, P.: Music mood representations from social tags. In: ISMIR, pp. 381–386 (2009)

[3] Çano, E., Morisio, M., *et al.*: Music mood dataset creation based on last.

**Table 7** Experiment results. Cells values correspond to the RMSE value.

| Experiment | Danceability | Acousticness | Energy | Valence | Instrumen |
|---|---|---|---|---|---|
| $Base_{t}oken_{weight}$ $XGBoost_{T}ags_{energy}$ | 300 | | | | |
| XGBoost - $y_{energy}$ | 300 | | | | |
| XGBoost - Tokens - Weight Repeat | 300 | | | | |

[1] Tolerance for the stopping criteria.

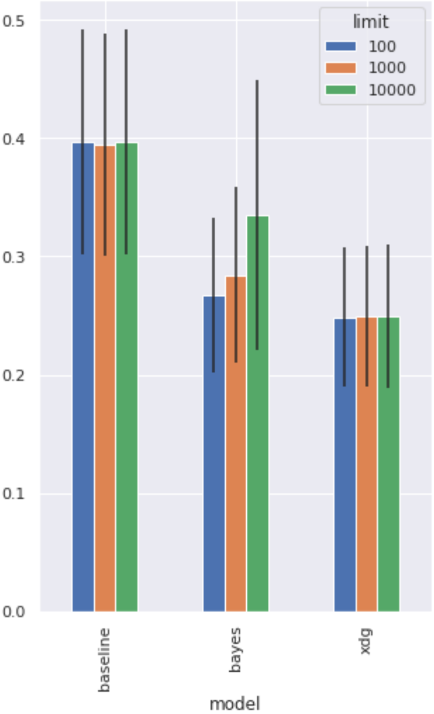

**Fig. 2** RMSE mean and standard deviation by model and tags/tokens limit.

fm tags. In: 2017 International Conference on Artificial Intelligence and Applications, Vienna, Austria, pp. 15–26 (2017)

[4] Bodó, Z., Szilágyi, E.: Connecting the last. fm dataset to lyricwiki and musicbrainz. lyrics-based experiments in genre classification. Acta Universitatis Sapientiae, Informatica **10**(2), 158–182 (2018)

[5] Bertin-Mahieux, T., Ellis, D.P.W., Whitman, B., Lamere, P.: The million song dataset. In: Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011) (2011)

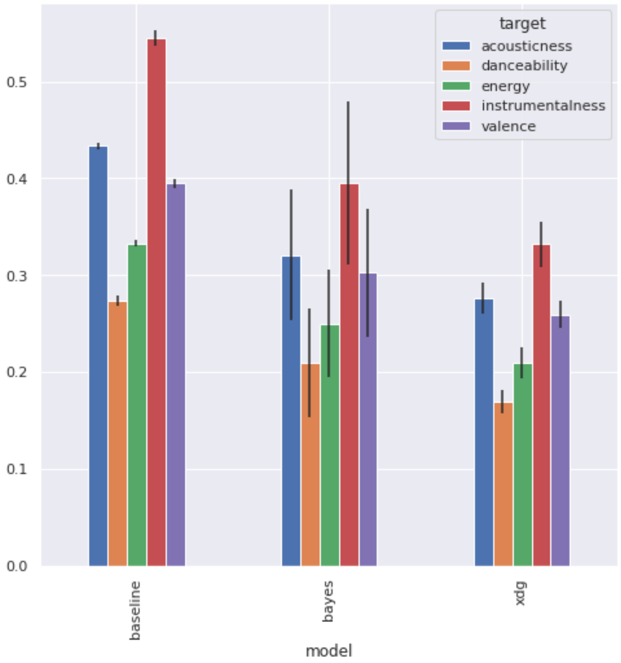

**Fig. 3** RMSE mean and standard deviation by model and audio feature.

[6] Wang, Y., Horvát, E.-Á.: Gender differences in the global music industry: Evidence from musicbrainz and the echo nest. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 13, pp. 517–526 (2019)

[7] Jamdar, A., Abraham, J., Khanna, K., Dubey, R.: Emotion analysis of songs based on lyrical and audio features. arXiv preprint arXiv:1506.05012 (2015)

[8] Benzi, K., Kalofolias, V., Bresson, X., Vandergheynst, P.: Song recommendation with non-negative matrix factorization and graph total variation. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2439–2443 (2016). Ieee

[9] Pinter, A.T., Paul, J.M., Smith, J., Brubaker, J.R.: P4kxspotify: A dataset of pitchfork music reviews and spotify musical features. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 14, pp. 895–902 (2020)

[10] xgboost: Xgboost. https://xgboost.readthedocs.io/en/stable/tutorials/model.html

[11] Tipping, M.E.: Sparse bayesian learning and the relevance vector machine.

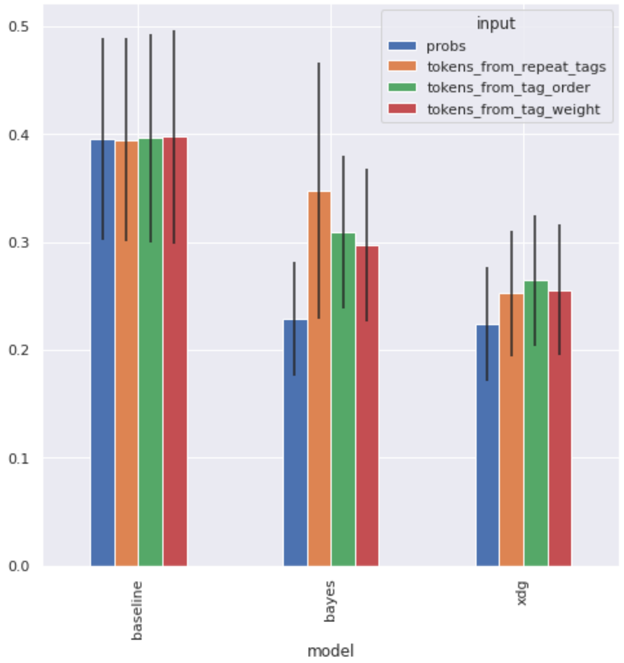

**Fig. 4** RMSE mean and standard deviation by model and input type (tag probablities or tokens).

J. Mach. Learn. Res. **1**, 211–244 (2001)

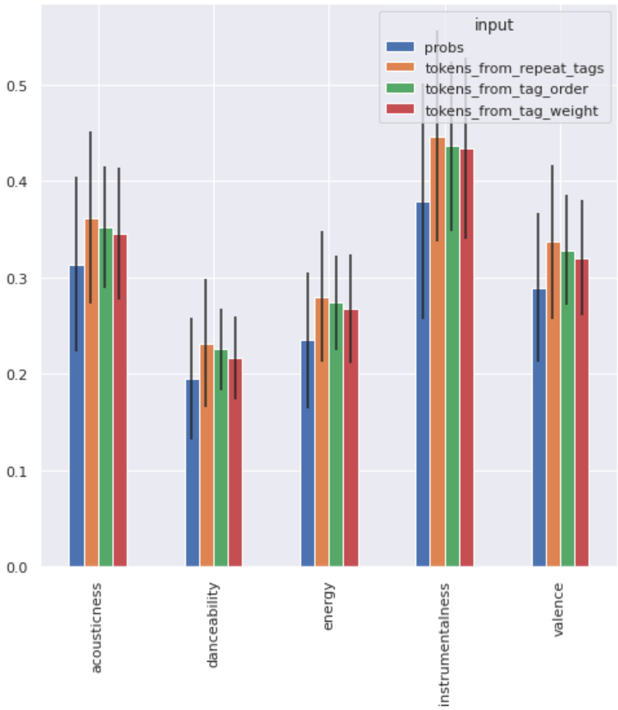

**Fig. 5** RMSE mean and standard deviation by audio feature and input type (tag probablities or tokens).