Automated Playlist Generation

KADE KEITH, Stanford University
DEMETRIOS FASSOIS, Stanford University

Our project generates music playlists based on a song or set of seed songs, using diverse features ranging from lyrical sentiment to song popularity. We approach the problem as both a graph problem and as a classification problem, and evaluate our results based on real human-curated playlists.

1 INTRODUCTION

TODO with the growth of music streaming services, there are now more songs than ever at music listeners fingertips. Because of this growth, the art of constructing playlists has become increasingly challenging, and discovering new music the in the expanse of choices is a daunting task. For this reason, we seek to build an automatic playlist generator, that can take a few songs as a seed set, and generate a complete playlist for the listener.

2 RELATED WORK

TODO

3 DATASET AND FEATURES

TODO

3.1 Feature Extraction

Apart from the audio features that were included in the million song dataset (acousticness, tempo, instrumentalness, liveness, speechiness, valence, danceability) we also extracted the year and popularity from the Spotify API data. We also crafted 3 additional features using the modeling techniques described below.

3.1.1 Latent Dirichlet Allocation. We chose all playlists for which we had an overlap of at least 30 songs with our dataset. We then tokenized and removed all stop words from the lyrics of the songs from every playlist, in order for the playlists to be treated as documents and the lyrics as words in the latent Dirichlet allocation model. Latent Dirichlet allocation is a generative statistical model that posits that the lyrics from every playlist can be explained by a fixed number of unobserved groups, which would explain similarity between some playlists. In our case, we chose the number of common topics to be 3, and the process that the generative model describes is the following:
For the M playlists, each of length N_i we have the following parameters and distributions:

(1) Probability $\theta_i \sim Dir(\alpha)$, where $Dir(\alpha)$ is a Dirichlet distribution with parameter α and $i \in 1, ..., M$
(2) Probability $\phi_k \sim Dir(\beta)$, where $k \in 1, 2, 3$ is the index of the topic.
(3) For each word in the lyrics from all playlists, for i, j, where $i \in 1, ..., M$ and $j \in 1, ..., N_i$:
   (a) Chose a topic $z_{i,j} \sim Multinomial(\theta_i)$
   (b) Chose a word $w_{i,j} \sim Multinomial(z_{i,j})$

The probabilities that were output from the model for each of the three topics, were subsequently used as features by the final model.

$$P(t) = \frac{b^{\frac{t+1}{T+1}} - b^{\frac{t}{T+1}}}{b-1}, \tag{1}$$

where $t = 0, \ldots, T$, and b is a number greater than 1.

3.1.2 Hidden Markov Model on timbre segments. Timbre is defined as the perceived sound quality of a musical note, sound or tone. The MSD dataset contains the time sequence of the timbre feature as a vector of 12 unbounded values centered around 0. These values represent different characteristics of the spectral surface, ordered by degree of importance. For example, the first dimension represents the average loudness of the segment, the second one describes brightness, the third one describes the flatness of a sound, the fourth describes sounds with a stronger attack etc. We averaged the vector features for every time segment for each song, in order to have a time series of the actual timbre of each segment. We subsequently trained a hidden Markov model on the timbre sequence of a random sample of 5,000 songs. The model is fit using the EM algorithm which is a gradient-based optimization method and can therefore get stuck in local optima. For this reason we fit the model with various initializations and selected the highest scoring one. The inferred optimal hidden states of the timbre segments of all songs were predicted by the model, employing the Viterbi algorithm. For each song the optimal hidden states were averaged to provide a single description of the path followed, which was used as an additional feature by the final model. For the training data the average value of the timbre which had a hidden value of 1 was 4.96, for hidden value of 1 they had an average value of the timbre of 13.76 and for a hidden value of 2 an average of -5.35.

3.1.3 Sentiment analysis. TODO

4 METHODS

Two different methodologies were pursued. In the first one we tried different classification algorithms both for a single playlist prediction and multiclass prediction for 26 playlists. In the second TODO

4.1 Classification

We first applied binary classification using a single playlist ('60s, 70s, 80s Classic Rock') as the target and randomly selecting songs that didn't belong to it as well. The training set consisted of 98 songs, while the test set included 34 songs. After standardizing the features we performed grid search on the

hyper-parameters for a logistic regression and support vector machine classifier. For logistic regression the regularization strength was fine tuned employing 10-fold cross-validation, while for the support vector classifier the regularization was also optimized using grid search. The other parameters that were chosen during cross validation were the kernel (linear or exponential). For the exponential kernel $e^{-\gamma\|x-x'\|^2}$ the γ parameter was also optimized.

4.2 Graph

5 RESULTS

Results for the two main methodologies followed are presented below.

5.1 Classification

The logistic regression model achieved training accuracy of 95.9% and CV accuracy of 94.8%. The SVC outperformed the logistic regression model, with the best model achieving training accuracy of 96.9% and CV accuracy of 95.9%. The best model's parameters used a linear kernel with regularization. The final test accuracy for the SVC was 94.1%. . The learning curve for the svm model is presented below in figure 1 and it shows that it generalizes well over unseen data.

The validation curve for the SVC model can be seen below in figure 2, which justifies how the particular value for the regularization hyper-parameter was chosen. The cross-validation accuracy decreases as the regularization parameter C increases even though the training accuracy keeps increasing, which would indicate overfitting.

The confusion matrix for the SVC model on the test data can also be seen below in figure 3.

The ROC curve is also plotted below in figure 4 which shows an excellent area under the curve.

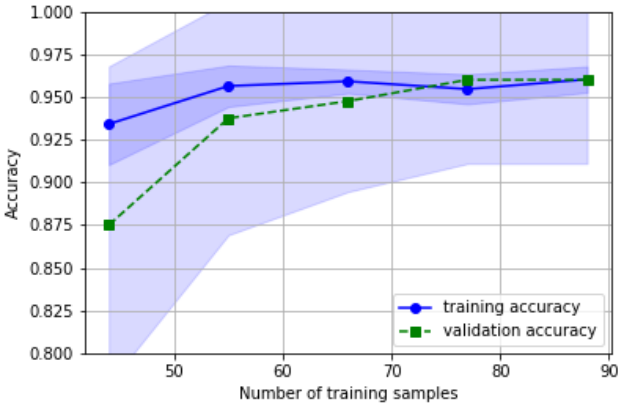

Fig. 1. Learning curve for SVC

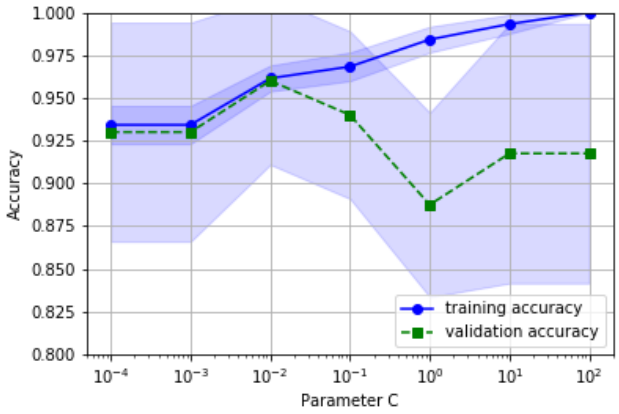

Fig. 2. Validation curve for SVC

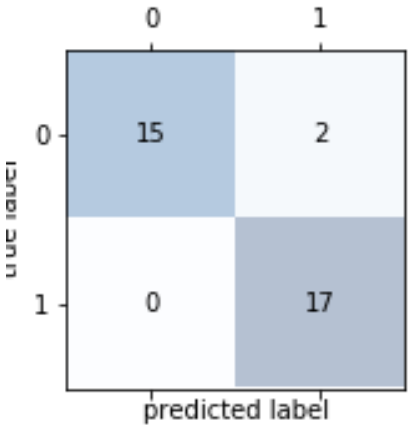

Fig. 3. Confusion matrix for SVC

Table 1. SVC performance on single playlist

Test accuracy	94.1%
Recall	100%
Precision	88.2%
F1-score	93.7%

Precision scores for SVC model

We also applied multi-class classification for 27 playlists for which we had more than 30 songs in our dataset. With this modeling technique called one-vs-all classification, one classifier is fitted for each class against all of the other classes. The training set consisted of 1265 songs, while the test set included 317 songs. Given the superior performance of the SVC model on one playlist, we optimized it in the multi-class setting using grid search and cross-validation.

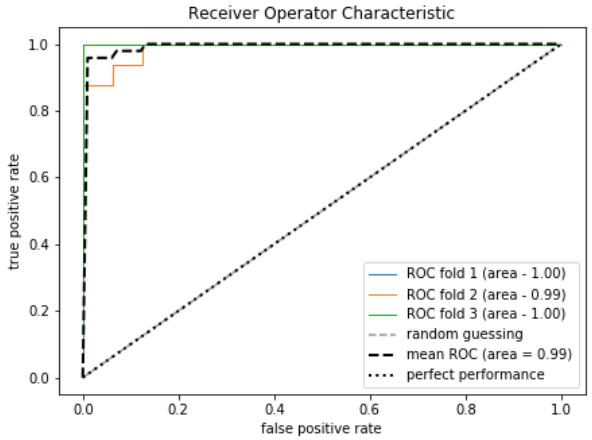

Fig. 4. ROC curve for SVC

5.2 Graph

TODO

Baseline: Avg distance within train set: 0.241749220274 Avg distance of prediction to positive train set: 0.191966891286 Avg distance of prediction to positive test set: 0.196542014849 Avg distance of prediction to negative test set: 0.292054955012

19.0 Avg distance to actual playlist: 0.231436866148

6 CONCLUSIONS

In this article, we develop the first multifrequency MAC protocol for WSN applications in which each device adopts a single radio transceiver. The different MAC design requirements for WSNs and general wireless ad-hoc networks are compared, and a complete WSN multifrequency MAC design (MMSN) is put forth. During the MMSN design, we analyze and evaluate different choices for frequency assignments and also discuss the nonuniform back-off algorithms for the slotted media access design.

7 TYPICAL REFERENCES IN NEW ACM REFERENCE FORMAT