Bayesify - A personal music taste analysis using Bayesian Networks

Napolitano Daniele
Master's Degree in Artificial Intelligence, University of Bologna
daniele.napolitano4@studio.unibo.it

March 10, 2023

Abstract

Modern streaming platforms are known for their ability to predict the preferences of their users. The music industry in particular poses a complex challenge due to the vastness of different music genres and songs available.
This project aims to model a Bayesian Network from a dataset built by fetching my musical data from Spotify's API, to experiment with different inferences and methods in order to find interesting relationships. Finally, the prototype of a use case application is presented.

Introduction

Domain

The dataset was built using my personal data taken from Spotify's API, and it is composed by various song properties.
The data has been cleaned and discretized, in order to ease the inference process. It is composed of 600 songs in total: 300 I like and 300 I dislike.

Aim

- Fetching data from API and performing preprocessing, to build a dataset which can be fed to a Bayesian model.
- Designing the structure of a causal network
- Trying automatic model generation from data, and comparing with the first one.
- Comparing different inference algorithms.
- Utilizing the acquired knowledge to develop a prototype of a practical use case.

Method

1. The dataset was built by fetching personal data from Spotify's API, cleaned and analyzed to find statistical correlations between features, and removing the irrelevant ones. The library Spotipy was used to make all the API requests.
2. Two different network structures were modeled: The first using domain knowledge and intuition[4], the other using *Hill Climb Search*[3] and BDeuScore [1] as scoring method.
3. PGMpy Python library was used to define the Bayesian network and performing inferences.

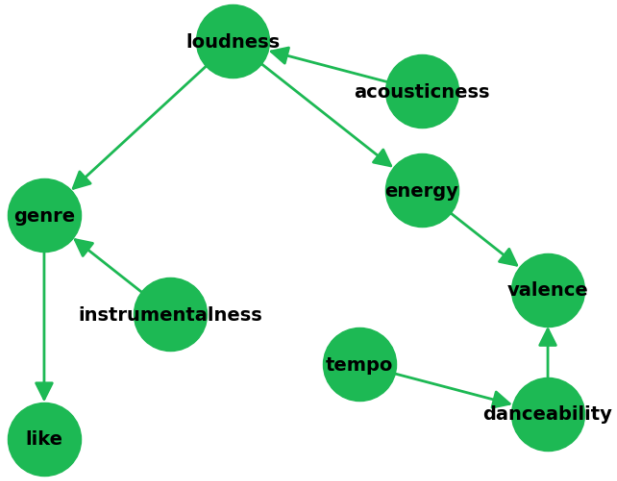

Figure 1: Bayesian Network based on domain knowledge.

4. *Maximum Likelyhood Estimator* was used to estimate probabilities from data, in order to provide the CPDs of the network.
5. Exact and Approximate inference were compared, using different methods for each approach, analyzing the results (i.e. Variable elimination and Belief Propagation for exact inference, and varying sample size for approximate inference).
6. A prototype of a recommendation system is made, where a song is fetched from Spotify and preprocessed in the same way of the training dataset, then inference is performed using all the feature but "like" as evidence, and the latter as target.

Results

The final model is able to provide good results when performing exact inference. On the other hand, approximate inference gave unexpected different results in some cases, while also taking more time to compute

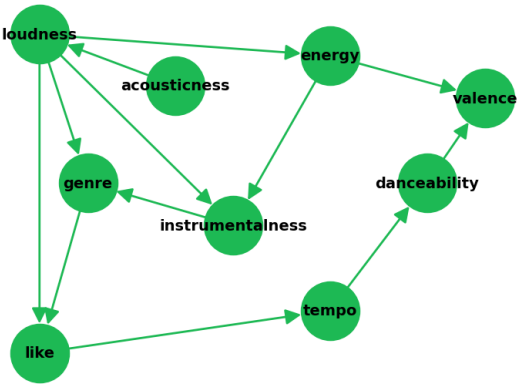

Figure 2: Bayesian Network made with Hill Climb Search.

Model

The following is a brief description of nodes in the model (Figure 1), partially taken from the original API documentation[2].

- **Like:** indicates whether I like a song or not (binary)
- **Genre:** a list of 10 possible genres.
- **Instrumentalness:** binary indicator of whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal".
- **Danceability:** describes how suitable a track is for dancing based on a combination of musical elements.
- **Energy:** represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy.
- **Loudness:** amplitude expressed in Decibel (dB).
- **Acousticness:** a confidence measure whether the track is acoustic.
- **Valence:** describes the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).
- **Tempo:** the overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.

With the exception of the first three features, all the others have values in range 0-4.
For both models, *Maximum Likely Estimator* was used to estimate the probabilities of the CPDs.

Results

The model that was built with Hill Climb Search (Figure 2) did not respect causality, linking "like" to other features. But still, it approximately behaved as the other model when comparing inference results.
Exact inference behaved as expected, confirming the strong relationship between "like" and "genre". Approximate inference gave unexpected results with the query $P(genre|like = 1)$, assigning higher values to pop music (especially when considering many number of samples). There seems to be a problem with sampling (PGMpy uses Rejection sampling).
The model I have built may represent correctly my personal musical tastes, but it might not perform as well with someone else's music data: the causal relationships I have modeled are based on my preferences, which are mostly *subjective* (i.e. some people might not care as much about genre, or only like happy songs, making the existing causal links not valid). The same thing can be said about the hill-climb model: it was generated by looking exclusively at my data, so it probably cannot generalize well with other people. Also, this model could not even work with myself in the future, as music taste changes over time.

Conclusion

The final models are simplistic (due to the discretization and naive assumptions when building the dataset) and subjective, but they still provided interesting insights on relationships between features. Experimenting with Bayesian Networks offered insights for advantages and disadvantages of each method. Additionally, building the dataset from scratch was useful to better understand the data, and really helped in the process of designing the causal network.

Links to external resources

The project was inspired by a Kaggle Dataset.
The link to Spotify's API documentation is: https://developer.spotify.com/documentation/web-api/
The whole project has been published on GitHub at the following link: github.com/danielenapo/Bayesify
The most relevant files are:

- *Dataset.ipynb:* notebook that generates the dataset, with all the explanations on how data was processed, and the reasons behind each choice.
- *Bayesify.ipynb:* main notebook with all the experiments described above in PGMpy.
- *spotifyData.csv:* the dataset used for this project (cleaned and discretized).

References

[1] Heckerman, D.; Geiger, D.; and Chickering, D. 1995. Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning* 20:197–243.

[2] Spotify. 2023. *Web API reference, Audio features*.

[3] Tsamardinos, I.; Brown, L.; and Aliferis, C. 2006. The max-min hill-climbing bayesian network structure learning algorithm. *Machine Learning* 65:31–78.

[4] Xiao-xuan, H.; Hui, W.; and Shuo, W. 2007. Using expert's knowledge to build bayesian networks. In *2007 International Conference on Computational Intelligence and Security Workshops (CISW 2007)*, 220–223.