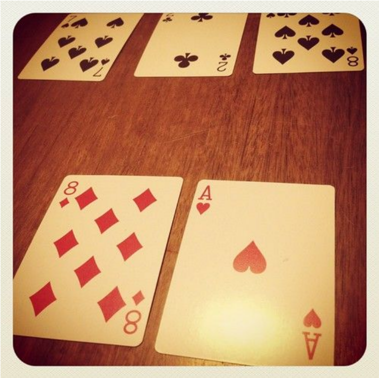


Song Twenty-One

To Listen or Not to Listen, That’s the Data-Driven Decision

Introduction and Motivation

Imagine that it is the due-date of an assignment deadline, and a college student wishes to take a break from their studies by listening to some music with the aim of getting an energy-boost from it. Inspired by the one-player version of the game twenty-one from the last course assignment, I will be modeling the decision of whether or not to play the next song as a Markov Decision Process (MDP).

The objective is for a college student to listen to songs during a study break on the due-date of an assignment, one at a time from a 40000+ song playlist and acquire a cumulative energy level as large as possible without going over the threshold value of 21, after which one is too distracted and energized to study, and then they cannot finish their assignment and they are "bust". This value is dependent on ‘energy' audio feature from Spotify's API.

Dataset Specifics

For my project, I have chosen The Spotify Hit Predictor Dataset (1960-2019), a dataset of 41106 songs and their audio features from Spotify's API ranging from 1960 to 2019 (across six decades). These audio features are very informative and range from danceability to key. Each song has a corresponding energy value between 0 and 1, that represents a perceptual measure of intensity and activity.

I chose the energy feature as through exploratory data analysis, I found that energy is positively linearly correlated with statistical significance ($p < 0.05$) to valence or emotions that represent happy, cheerful songs closer to 1 and sad songs closer to 0, and I wanted to find a feature that would represent a mood-boost after listening to it during the study break. I discretized this into 10 energy levels.

Typically, overly energetic tracks feel fast, loud and distracting, and thus, there is a threshold value for energy level assumed as 21 (like twenty-one game) in cumulative energy level since then the student will get too energized and distracted and "bust" because they are likely not to finish the assignment by the deadline.

Background Continued

At each turn in the study break, the college student has one of two actions: "play" to play the next song, and "stop" to stop listening to music. We will only study scenarios in which the cumulative energy level is greater than 11, since the optimal decision is always to play the next song otherwise (since each song can at maximum have a energy level of 10, and the student won't get distracted, and they will go back to studying, hence finishing the assignment and not going "bust").

If the student has a late day for the assignment, that gives them the power like a usable ace in the game twenty-one not to get distracted and possibly fail to submit the assignment in time (not to go "bust"). Just like a usable ace only in function (but not in any other likeness), we let the late day have either value 1 or 11, as long as the cumulative energy level is not greater than 21 (going "bust"). We call the latter situation a "late day" and track this as part of the state.

Overview of Methods

Step 1: Defining the Markov Decision Process (MDP)

- States: Tuples [0,1]x[12,...,21] to represent Yes/No late day and Cumulative Energy level >= 12 ; Actions: Stop (0) and Play (1)

Step 2: Constructing a 20x20 Transition Matrix for the “Play” Action

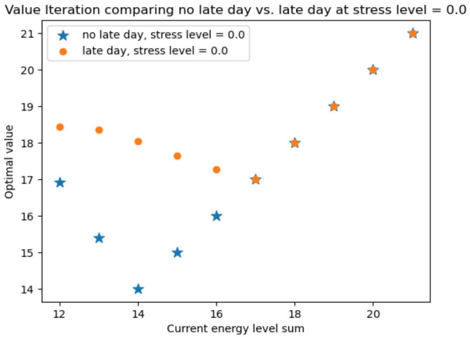
- The transition matrix T for the "play" action can be defined as $T_{ij} = P(j|i)$ and i and j index the states as described above. Each row corresponds to a current cumulative energy level state, and each column corresponds to the next cumulative energy level state after playing a song.

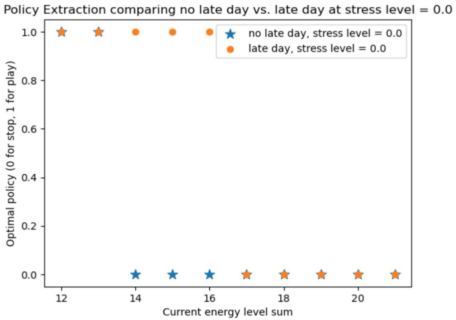
Step 3: Finding Optimal Actions through Value Iteration and Policy Extraction

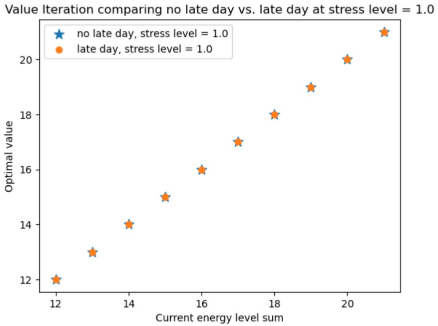
- Inputs: transition matrix, baseline stress level factor (1- discount factor), and convergence threshold. Output: 2 length-20 vectors of converged optimal values and policies.

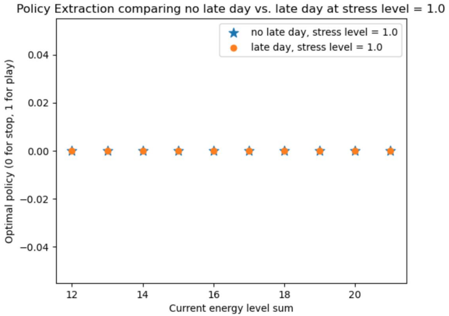
Results

In order to analyse optimal decision making through value iteration, I ran instances of value iteration with the following values of baseline stress level: eta=0, eta=0.1, eta=0.5 and eta=1.0. For each one, I produced two scatter plots showing the values and policies of the states 12-21, overlaying both the sets of values/policies without and with a late day on the same plot









Low stress level = 0 (in general <= 0.5)

High stress level = 1 (in general >= 0.5)

Analysis of Results

- The optimal actions in states with no late day are in to play next song until cumulative energy level 13, and then stop (for low stress level) but to always stop (for higher stress level). These actions do depend on the value of baseline stress level eta.
- For eta=0, the optimal actions in states with a late day are to play the next song up until the sum of 16 and stop listening to a music for a sum >= 17. For eta=0, no usable late day and energy level sums >= 12, the optimal action is play the next song up until the sum of 13 and stop listening to a music for a sum >= 14. The reason for the discrepancy in optimal values is because the college student is optimistic under uncertainty when they have a late day (as a kind of a veto power) and wants to play next song in their study break so they can possibly get a higher energy boost in in the future.
- The optimal strategy in the case of a late day changes as we increase eta, wherein we stop listening to music earlier as baseline stress levels are higher

Limitations and Future Work

In terms of limitations, I believe that the single feature of energy may be too little to score a song by, and I could try to leverage many other brilliant features from the dataset like danceability, valence, loudness and so on. Further, the transition matrix probabilities in my project are constructed from data of all 40000+ songs, but should probably change to using a smaller subset like a user-defined playlist or a group of recommendations or songs by the same artist.

For future work, I would like to make a simple app called 'Song Twenty-One' with a UI that simulates this decision making process like an interactive game for college students, and further I would like to create the basic Music Recommendation System based on K-Nearest Neighbors for a song feature vector and a chosen metric, modifying the transition matrix with a smaller chosen song subset.

Thank you!