**1. Content-Based Filtering Overview**

Content-based filtering is a recommendation technique that suggests items similar to those the user has liked before. Unlike collaborative filtering (which relies on user behavior and preferences from a community), content-based filtering focuses on the properties or "features" of the items themselves.

In the case of a music recommendation system, this means analyzing various attributes of songs (such as genre, tempo, key, mood, etc.) and using them to recommend similar tracks.

**Key Techniques**

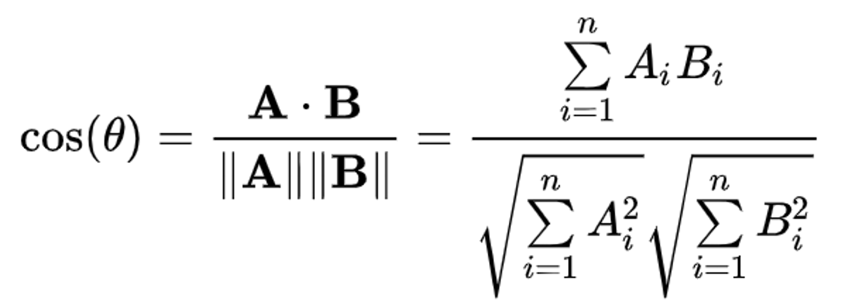
**a) Feature Extraction**

* Songs are broken down into numerical features that describe them.
* Common song features include:
  + **Genre:** e.g., pop, rock, hip-hop
  + **Tempo (BPM):** speed of the song
  + **Key and Mode:** musical key (C major, D minor, etc.)
  + **Energy:** perceived intensity and activity
  + **Danceability:** how suitable a track is for dancing
  + **Valence:** musical positiveness
  + **Duration:** length of the track
  + **Acousticness, Instrumentalness, Liveness:** audio characteristics

Data can be sourced from platforms like **Spotify’s Web API**, which provides song-level metadata.

**b) Cosine Similarity**

* A widely used method to measure similarity between two songs based on their feature vectors.
* Calculates the cosine of the angle between two feature vectors.
* The closer the cosine value is to **1**, the more similar the songs are.

**Formula:**

**Example Workflow:**

1. User inputs songs they like.
2. Extract features of those songs using APIs or datasets.
3. Represent each song as a vector of features.
4. Compute cosine similarity between liked songs and other songs in the dataset.
5. Recommend the most similar songs based on the highest similarity scores.

**2. Using Song Features for Recommendations**

By comparing songs based on the above features, you can create highly personalized suggestions. For instance:

* If a user likes high-energy pop songs with a tempo between 120-140 BPM and major key, recommend tracks with similar attributes.
* You could also add user-defined weights to certain features (e.g., prioritize "genre" over "tempo") to fine-tune recommendations.

You might also combine this with **normalization** techniques (e.g., Min-Max Scaling) to ensure features on different scales don’t dominate the similarity calculation.

**Tools/Libraries:**

* **Python Libraries:**
  + scikit-learn for cosine similarity & normalization
  + pandas for data manipulation
  + spotipy (a Python wrapper for Spotify API)

**Ideas:**

* We could integrate Spotify’s API early on to extract real-world song data.
* Potential to enhance with user-adjustable filters (e.g., “prefer danceable tracks” slider).
* Down the line, hybridize with collaborative filtering to boost recommendation accuracy.

**A computer screen shot of a program

AI-generated content may be incorrect.Example** in python:

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.