## Robert Brod

## 001939918

## 

## C964: Computer Science Capstone

## Task 2: Parts A, B, C, and D

### 

### Part A: Project Proposal for Business Executives

#### Letter of Transmittal

10/6/2023

Song Recommendation Project

Dear Djentbox Senior Leadership,

In today’s digital age, the vast array of music choices often overwhelms music enthusiasts, making it challenging for them to discover new songs that truly resonate with their tastes and moods. This project aims to address this issue by developing an advanced recommendation program that uses cutting-edge machine learning techniques and rich song data to provide users with highly personalized and enjoyable music recommendations.

I recommend the creation of a data product, the Djentbox Song Recommendation program, as the solution to the problem. This data product will utilize a combination of machine learning, data analysis, and advanced visualization techniques to cluster songs based on their inherent features and provide users with precise and enjoyable music recommendations, thereby enhancing their overall music-listening experience.

Implementing the Djentbox Song Recommendation program will offer several key advantages to Djentbox, including increased user engagement and satisfaction. This data product has the potential to extend user sessions, boost user loyalty, and differentiate Djentbox in the competitive music streaming market. Additionally, it will enable us to gather valuable insights into user preferences and behavior, allowing for more data-driven decision-making and product enhancements.

In terms of funding, our project requires a comprehensive allocation to support key areas. This includes approximately $350,000 for development environments and hardware resources, $800,000 for personnel salaries and benefits, $200,000 for essential software tools and libraries, and $75,000 for ongoing maintenance and support. These allocations ensure that we have the necessary resources to create, maintain, and continuously enhance our music recommendation system, delivering an outstanding user experience.

As the individual leading this project, I want to assure you of my qualifications for this endeavor. My educational background in computer science has provided me with a strong foundation. Furthermore, I have practical experience in working with various programming languages, database management systems, and user interface design, all of which have prepared me to effectively develop and deliver the Djentbox Song Recommendation program.

Best regards,

Robert Brod

#### 

#### Project Recommendation

**Problem Summary**

The Djentbox Song Recommendation project is poised to transform our music recommendation capabilities. Leveraging a strong background in computer science, I will develop this data product to tackle the challenge of overwhelming music choices faced by our users. I will create a recommendation engine that clusters songs based on their characteristics, ensuring users receive personalized music suggestions. This initiative will also feature user-friendly visualizations and an intuitive recommendation approach. The Djentbox Song Recommendation project promises to boost user engagement, satisfaction, and loyalty, positioning Djentbox as a leader in the music streaming space.

With the dynamic landscape of music streaming, Djentbox has emerged as a platform committed to providing unparalleled music experience to its users. With an extensive and diverse catalog spanning countless genres, artists, and tracks, Djentbox offers music enthusiasts access to a treasure trove of musical content. However, this wealth of musical options, while a testament to our commitment to variety and choice, can sometimes overwhelm our users. Despite the effectiveness of our current music recommendation system, users may still find it challenging to navigate this vast catalog and discover new songs that resonate with their preferences. Recognizing this need, I am introducing the Djentbox Song Recommendation project to Djentbox. This endeavor arises from the necessity to refine and personalize our music recommendation system further. Leveraging the cutting-edge capabilities of data science and machine learning, this project aims to elevate our recommendation capabilities, ensuring that Djentbox users not only receive relevant music suggestions but also enjoy an enriched and tailored music-listening experience. In this highly competitive music streaming industry, the Djentbox Song Recommendation project serves as a strategic response to the demand for superior user experiences and reaffirms Djentbox’s commitment to being a trailblazer in the field.

The Djentbox Song Recommendation project aligns perfectly with Djentbox’s strategic goals and user-centric approach. By enhancing our music recommendation capabilities, this project directly addresses the need for a more refined and personalized user experience. It ensures that our users receive tailored music suggestions, reducing choice overload and increasing engagement. This aligns with Djentbox’s mission to provide unparalleled music experiences, ultimately strengthening user satisfaction and loyalty. Additionally, the project offers valuable insights into user preferences, enabling data-driven decision-making to further optimize our services and maintain our competitive edge in the music streaming industry.

Through this data product, we will deliver a cutting-edge music recommendation system that leverages data science and machine learning to provide personalized song suggestions to Djentbox users. This system will include a refined recommendation algorithm, user-friendly visualizations for non-technical leadership to explore, and an intuitive user interface.

Djentbox users will benefit from a more tailored and enjoyable music-listening experience, leading to increased engagement and satisfaction. The recommendation system will help users navigate Djentbox’s extensive music catalog, reducing the feeling of choice overload.

The project will provide valuable insights into user preferences, enabling data-driven decision-making for future platform enhancements. By staying at the forefront of music recommendation technology, Djentbox will maintain a competitive edge in the music streaming industry. The refined recommendations and improved user experience will lead to higher user loyalty and retention rates, contributing to Djentbox’s long-term success.

Overall, the Djentbox Song Recommendation project is poised to deliver a valuable and innovative solution that aligns with Djentbox’s mission and strategic objectives while meeting the evolving needs of our users in the ever-competitive music streaming landscape.

**Application Benefits**

Djentbox’s success relies on user engagement and satisfaction. The project addresses this need by offering personalized music recommendations, reducing the frustration associated with navigating a vast music catalog. Users will be more likely to spend additional time on the platform, explore more content, and, as a result, have a higher level of satisfaction.

In the competitive music streaming industry, user loyalty is paramount. By consistently delivering music recommendations that align with user preferences, the project will foster stronger user loyalty. Satisfied users are more likely to remain active subscribers, reducing churn rates and contributing to long-term revenue growth.

By embracing advanced data science and machine learning techniques, Djentbox distinguishes itself from competitors. Offering a more refined music recommendation system positions Djentbox as an industry leader in innovation and user experience, attracting new users and retaining existing ones.

A more engaged and satisfied user base can open avenues for monetization beyond subscription fees. This includes partnerships, targeted advertising, and premium content offerings, all of which can contribute to revenue diversification and growth.

The project is designed with scalability in mind. As Djentbox’s music catalog and user base expand, the recommendation system will adapt, ensuring consistent performance and user satisfaction. This futureproofing is essential for sustained business success in a rapidly evolving industry.

Djentbox’s mission is to provide exceptional music experiences to its users. The Djentbox Song Recommendation project aligns seamlessly with this mission, reinforcing the organization’s commitment to delivering value and joy to its users.

In summary, the Djentbox Song Recommendation project offers multifaceted benefits that directly address Djentbox’s business needs. It not only enhances user engagement and satisfaction but also provides valuable data insights, fosters loyalty, and strengthens Djentbox’s competitive position in the music streaming market. This initiative represents a strategic investment in the long-term growth and success of Djentbox.

**Application Description**

At its core, the application employs the K-Means Clustering algorithm to categorize songs into clusters based on their inherent characteristics, ensuring that users receive music suggestions aligned with their preferences. Principal Component Analysis (PCA) further simplifies the understanding of song clusters by reducing dimensionality and creating a 2D scatter plot that provides an intuitive visualization of the recommendation process.

The recommendation engine calculates the ‘distance to cluster’ for input songs and identifies songs within the same cluster with the closest distances, offering users recommendations that not only share similar features but also provide a cohesive listening experience.

While users may not directly interact with the visuals, the application generates informative visualizations using the Matplotlib library. These visuals are intended to provide non-technical members of leadership with insights into the recommendation process and data-driven decision making. Matplotlib, a powerful data visualization library in Python, plays a crucial role in creating these visuals. Specifically, Matplotlib enables the generation of a 2D PCA reduction scatter plot, a danceability value variation bar chart, and feature-specific boxplots. The 2D PCA reduction scatter plot allows for an intuitive understanding of song clusters, while the danceability value variation bar chart offers a clear representation of how a feature varies among recommended songs. Additionally, the feature-specific boxplots visually summarize statistics for each feature, providing a comprehensive overview of the recommendation data.

To ensure consistent performance and scalability, the application is engineered to accommodate Djentbox’s growing music catalog and user base without compromising the quality of recommendations. This technical approach is tailored to meet Djentbox’s business needs by enhancing user engagement, satisfaction, and loyalty through a refined and data-driven music recommendation system.

**Data Description**

The raw data utilized in the Djentbox Song Recommendation project originates from Kaggle, a well-regarded platform known for hosting diverse and comprehensive datasets. Kaggle provides a rich repository of high-quality datasets, and in this case, it serves as the primary source of song-related data, including features such as danceability, energy, speechiness, acousticness, instrumentalness, liveness, and valence. This dataset serves as the foundation upon which our recommendation system is built, enabling us to leverage the power of data science and machine learning to enhance the music recommendation experience for Djentbox users.

The dataset comprises various columns, each serving specific purposes in the Djentbox Song Recommendation project. Four of the columns, namely “ID,” “Artist,” “Track,” and “Album,” fall under the nominal category. These columns function as categorical variables, as they categorize songs, artists, tracks, and albums without any inherent order or numerical meaning. They play vital roles in identifying and categorizing songs and their associated metadata.

The dataset also includes seven columns, “Danceability,” “Energy,” “Speechiness,” “Acousticness,” “Instrumentalness,” “Liveness,” and “Valence,” which serve as the primary audio features used for recommendation. These features are quantitative, characterized by continuous values ranging from 0.0 to 1.0. For instance, “Danceability” quantifies the dance-friendliness of a song, with a value of 1.0 indicating high danceability, while a value of 0.0 suggests the opposite. These quantitative features are essential in the data analysis process, allowing for precise numerical comparisons and calculations.

Together, these columns provide the necessary data for the Djentbox Song Recommendation project, facilitating the development of a sophisticated recommendation system based on data science and machine learning techniques.

The dependent variable in this project is the recommended song. It represents the outcome or target variable that the recommendation system aims to predict or generate based on various factors.

The independent variables encompass a set of audio features and song-related attributes, including danceability, energy, speechiness, acousticness, instrumentalness, liveness and valence. These independent variables serve as input features for the recommendation system. They play a fundamental role in clustering songs and calculating the distance to cluster value for input songs, ultimately determining which songs to recommend to users based on their preferences and the similarity of these attributes. Together, these variables form the foundation of our data-driven approach to providing personalized music recommendations within the Djentbox platform.

In the Djentbox Song Recommendation project, we addressed data anomalies by removing rows with missing data (NaN values) during preprocessing, but potential data incompleteness beyond these rows remains a concern. The dataset might not fully represent all music genres and styles, affecting the recommendation system’s effectiveness, particularly for less-represented categories.

The accuracy of audio features depends on data sources and feature extraction methods, potentially impacting recommendation reliability. Our system assumes that clustering songs based on audio features alone results in suitable recommendations, but musical preferences are multifaceted. As the dataset grows, maintaining computational efficiency becomes challenging, necessitating ongoing optimization.

**Objectives and Hypothesis**

The Djentbox Song Recommendation project is driven by a set of key desired outcomes. The project aims to enhance the overall user experience, providing personalized music recommendations that resonate with users’ preferences. By doing so, it intends to increase user retention, reducing churn rates and fostering loyalty among subscribers.

Furthermore, the project seeks to boost user engagement by encouraging active music exploration and discovery. It aspires to provide valuable data-driven insights by leveraging user data for informed decision making.

The project emphasizes scalability, ensuring that it can efficiently accommodate an expanding music catalog and user base, thereby supporting Djentbox’s long-term growth in the industry.

In summary, these outcomes collectively contribute to the project’s overarching goal of delivering outstanding music experiences, fostering user loyalty, and positioning Djentbox as a prominent player in the music streaming landscape.

The hypothesis for this project is that by implementing a sophisticated recommendation system based on user preferences and audio feature analysis, Djentbox can significantly increase user engagement and retention, leading to reduced churn rates and ultimately strengthening its position as a leading music streaming platform in the industry.

The success of this project will be assessed based on a combination of metrics and user feedback. The project aims to increase user interaction with recommended songs, including playbacks, likes, shares, and playlist additions. Success will be reflected in elevated user engagement levels, indicating that users find value in the recommendations.

A reduction in churn rates is a key indicator of success. The project seeks to foster user loyalty, encouraging subscribers to remain with the platform for longer periods.

These evaluation criteria provide a comprehensive view of the recommendation system’s effectiveness and its impact on user behavior, satisfaction, and platform performance. Success will be defined and refined based on these real-world observations and user-centric metrics.

**Methodology**

In developing the Djentbox Song Recommendation project, I have chosen to adopt an Agile development methodology. This decision is based on several key factors that make Agile particularly well-suited for this project.

Agile places a strong emphasis on user feedback and involvement throughout the development process. Given that the success of our project hinges on delivering personalized and satisfying music recommendations, this user-centric approach is paramount. Agile allows us to continuously gather user feedback, ensuring that the recommendation system aligns closely with user preferences.

Agile methodologies are inherently flexible and adaptive. In the dynamic landscape of music preferences and streaming platforms the ability to respond to changing user needs and market trends is crucial. Agile’s iterative nature allows us to adjust and refine as we progress, ensuring that recommendations remain relevant and engaging.

Our Agile development approach will follow the Scrum framework, a widely recognized and effective Agile methodology.

In the initiation phase, we define the project scope, objectives, and initial user stories. We establish a project team, including developers, data scientists, designers, and user experience experts. Key stakeholders are identified, and initial backlog items are created. At the beginning of each sprint, the team will conduct sprint planning sessions. During these sessions, we will select high-priority backlog items for implementation during the sprint. The team outlines sprint goals and tasks.

The development phase involves coding, data analysis, and algorithm refinement. Continuous integration and automated testing ensure the quality of our software. Daily standup meetings facilitate communication and address challenges promptly. At the end of each spring, we conduct UAT sessions. Users validate and provide feedback on the implemented features and recommendations. This iterative process allows us to refine our recommendations based on user preferences.

After each sprint, retrospective meetings are held to reflect on the sprint’s successes and areas for improvement. Team members discuss what went well, what could be done better, and adjust to optimize performance in subsequent sprints. The project proceeds through multiple iterations, building upon previous ones. This iterative approach ensures that user feedback is continuously incorporated, and the system evolves to meet evolving requirements and user expectations.

**Funding Requirements**

The development environment includes cloud-based servers, databases, and storage solutions essential for development, testing, and deployment. Estimated budget: $250,000

Hardware Resources: High-performance servers and computing clusters will support data analysis and algorithm training. This includes server hardware, networking equipment, and related infrastructure. Estimated budget: $100,000.

The project team consists of developers, data scientists, designers, user experience experts, and project managers. Competitive salaries, benefits, and payroll taxes are included. Estimated budget: $800,000.

Various software tools and libraries are required for data analysis, algorithm development, user interface design, and software development. This budget covers licenses and subscriptions. Estimated budget: $200,000

Ongoing maintenance includes expenses related to maintaining and supporting the system after deployment. It covers activities such as bug fixes, updates, server maintenance, and user support. Estimated budget: $75,000.

In summary, the funding requirements for these categories are as follows:

* Environment: $350,000
* Personnel: $800,000
* Tools and Software: $200,000
* Maintenance and Support; $75,000

**Data Precautions**

Public datasets on platforms like Kaggle are typically curated and made available for public use, research, and analysis. They are often anonymized and do not contain personally identifiable information (PII) or sensitive data that would require special handling, privacy considerations, or regulatory compliance.

Since the dataset we are utilizing for this project is publicly accessible and does not involve sensitive or protected information, we can focus our efforts on data analysis, algorithm development, and user experience enhancements without the need for elaborate privacy measures or data security precautions that would typically be associated with sensitive data handling.

This simplifies the project’s data management and allows us to concentrate on delivering a robust and user-friendly music recommendation system while adhering to best practices for data analysis and model development.

**Developer’s Expertise**

I hold a bachelor’s degree in computer science, providing a strong foundation in software development and data analysis. Proficient in Python, I leverage libraries like Pandas, scikit-learn, and Matplotlib effectively. My skills encompass data analysis, machine learning techniques, and practical software development.

In addition, I’ve managed solo projects, demonstrating my ability to plan and deliver technical solutions. I’m well-versed in multiple programming languages, database systems, and user interface design.

My experience extends to leveraging external datasets, such as Kaggle’s, to derive actionable insights, emphasizing a commitment to data-driven solutions. These qualifications position me to lead the Djentbox Song Recommendation project, aligning with its objectives and ensuring a captivating music recommendation system.

My academic background in computer science, complemented by expertise in Python and relevant libraries, directly addresses the technical requirements of the project. This foundation allows me to conduct in-depth data analysis, apply machine learning techniques, and develop the essential algorithms needed for an effective song recommendation system.

Additionally, my proficiency in software development bridges the gap between data analysis and practical application. I can seamlessly translate complex analytical results into user-friendly features, ensuring that the project delivers tangible value to users.

Managing solo projects demonstrates my capability to independently plan, execute, and deliver technical solutions – a crucial aspect when overseeing the development of the recommendation system. My familiarity with diverse programming languages, database systems, and user interface design complements the multidisciplinary nature of the project, enabling me to collaborate effectively with the team.

Moreover, my experience in leveraging external datasets aligns with the project’s data-driven approach. It underscores my ability to extract valuable insights and refine recommendations based on user preferences, enhancing the overall quality of the music discovery experience.

### Part B: Project Proposal

**Problem Statement**

In the ever-evolving landscape of music streaming platforms, the challenge of providing personalized and engaging music recommendations is more pressing than ever. Despite the abundance of available songs, users often struggle to discover music that resonates with their individual tastes and preferences. Existing recommendation systems, while functional, often fall short in delivering truly tailored music suggestions.

The problem lies in the complexity of musical preferences. Songs aren’t just data points; they embody unique combinations of attributes such as danceability, energy, acousticness, and more. Traditional recommendation systems rely on simplistic models that fail to capture the nuanced relationships between songs and user preferences.

Furthermore, users seek a seamless and immersive experience. They want more than just a list of song titles; they desire a user interface that not only provides recommendations but also allows for exploration and interaction with the data behind the suggestions.

To address these challenges, we propose the Djentbox Song Recommendation project. Our goal is to develop a cutting-edge music recommendation system that leverages advanced data analysis techniques, machine learning algorithms, and interactive user interfaces to deliver highly personalized and engaging music discovery experiences to our users. By tackling these issues head-on, we aim to revolutionize the way users discover and enjoy music on our platform.

**Customer Summary**

Our primary target for the data product is Djentbox, a technologically advanced music streaming platform. Djentbox provides innovative music experiences to a diverse audience of music enthusiasts.

The audience for Djentbox includes music enthusiasts who span various genres, from pop and rock to classical and electronic. These users are passionate about discovering new music that aligns with their unique tastes. Additionally, Djentbox attracts tech-savvy users who appreciate advanced features and a high level of technical sophistication. There’s also a growing subset of data-driven listeners who seek transparent, data-backed recommendations. Lastly, Djentbox serves interactive experience seekers who value not only personalized song recommendations but also interactive exploration of music data.

Our proposed data product is poised to revolutionize music recommendation by leveraging advanced data analysis, machine learning, and interactive user interfaces. Traditional recommendation systems often rely on simplistic models that struggle to capture the complexity of musical preferences. Our data product employs advanced data analysis techniques that delve deeper into the nuanced attributes of songs. By considering factors such as danceability, energy, speechiness, acousticness, instrumentalness, liveness, and valence, we create a richer and more accurate understanding of each song.

We incorporate state-of-the-art machine learning algorithms, including K-means clustering, to group songs based on their unique characteristics. This allows us to identify subtle patterns and associations that go beyond genre labels or popularity metrics. As a result, our recommendations are not limited by genre constraints and offer a more personalized experience.

Data-driven listeners appreciate transparency in how recommendations are generated. Our data product will not only deliver personalized song suggestions but also provide users with insights into why a particular song is recommended. By showcasing the distance to cluster value, users gain a clear understanding of the rationale behind each recommendation, enhancing their trust in the system.

Interactive experience seekers will find our data product engaging and visually stimulating. The ability to interact with music data in a meaningful way sets Djentbox apart as a platform that offers more than just a playlist. It will offer an immersive and informative music discovery experience.

In summary, the Djentbox Song Recommendation data product will successfully resolve the problem by embracing data-driven innovation. Our advanced data analysis, machine learning algorithms, and interactive user interfaces will create a unique and engaging music recommendation system that goes beyond traditional models. By addressing the complexities of musical preferences and providing transparency, our data product is poised to deliver highly personalized and satisfying music discovery experiences to Djentbox’s diverse and tech-savvy user base.

**Existing System Analysis**

At present, Djentbox employs a rudimentary recommendation approach that primarily relies on suggesting songs from the same artist or album. This approach, while straightforward, lacks the sophistication required to provide a diverse and engaging music discovery experience.

Djentbox’s existing system primarily revolves around artist-centric recommendations. When a user listens to a song by a particular artist, the system recommends other songs by the same artist. While this approach can be effective for die-hard fans, it falls short in introducing users to a broader range of music.

In addition to artist-centric recommendations, Djentbox also suggests songs from the same album when a user selects a track. This strategy assumes that if a user enjoys one song from an album, they may like others from the same album. However, it neglects the diversity of user’s musical tastes and potential interest in songs from different albums.

The current system lacks true personalization and fails to consider users’ individual preferences, moods, or past listening history. Users receive recommendations based solely on the specific artist or album they are currently exploring, leading to repetitive and predictable suggestions.

Djentbox’s user interface is simple and static, providing basic lists of songs based on the user’s current selections. Users have minimal opportunities for interactive exploration or discovery beyond artist and album associations.

There is no data analysis or machine learning involved in the current recommendation process. The system relies solely on static associations between artists and albums, resulting in recommendations that lack depth and variety.

This overly simplistic recommendation approach limits Djentbox’s ability to offer a dynamic and engaging music discovery experience. To meet the evolving expectations of its users and compete effectively in the music streaming industry, Djentbox recognizes the necessity of transitioning to a more advanced and data-driven music recommendation system.

**Data**

The Djentbox Song Recommendation project will leverage a comprehensive music data set retrieved from Kaggle, a reputable data science community and platform.

The data set includes essential information about songs, including song titles, artist names, and album names. This metadata forms the foundational structure of our music database. For each song, a rich set of audio features is provided, offering insights into their musical characteristics. These features include danceability, energy, speechiness, acousticness, instrumetnalness, liveness, and valence. These features are represented as values ranging from 0.0 to 1.0, providing a granular view of each song’s attributes.

The Kaggle data set consists of a substantial volume of songs, over 20,0000, ensuring diversity and depth for accurate recommendations. With tens of thousands of songs, it forms a robust foundation for our recommendation system.

In the design phase we will define the data model that represents how music data will be structured and organized. This model will include classes/objects that hold data such as song name, artist, album, and features. We will design the schema for data storage, considering how data will be stored in Pandas DataFrames within the Jupyter Notebook environment. This schema will determine how data is organized and related.

In the development phase data will be acquired from the Kaggle data set and loaded into Pandas DataFrames. This phase will include extracting data from CSV format into DataFrames. Using Pandas, we will perform data preprocessing tasks such as data cleaning, feature engineering, and handling missing values. This ensures that the data used in the recommendation model is of high quality. Within the Jupyter Notebook environment, a machine learning model for recommendation will be developed, leveraging Pandas for data input and output during model training and evaluation.

During the testing and quality assurance phase data validation will be a critical component. We will use Pandas for data validation to ensure that the processed data aligns with expected standards and does not introduce errors into the system.

In the deployment phase scalability will be evaluated. If necessary batch processing techniques will be introduced within Pandas to handle increasing data volumes efficiently.

During the maintenance phase regular data backups and disaster recovery plans will be in place to prevent data loss and ensure uninterrupted service. Maintenance includes keeping data up-to-date. We will use Pandas to manage the ingestion of new data and update recommendation models as needed.

During the continuous improvement phase the data quality and system performance will be continuously monitored using Pandas to identify any anomalies or issues that may require attention.

**Project Methodology**

The Djentbox Song Recommendation project will be executed following the Agile methodology, an industry-standard approach that emphasizes flexibility, collaboration, and iterative development. Agile is particularly well-suited for dynamic projects where requirements may evolve over time. The development process will be divided into sprints, each typically lasting two weeks.

In the initial phase, the project team will define the scope, objectives, and key requirements of the recommendation system. This includes understating user expectations, defining the data sources, and outlining high-level features. The development environment within Jupyter Notebook will also be set up.

During the sprint planning phase, the team will break down the project into smaller tasks and prioritize them based on user value and complexity. Detailed designs for the recommendation system’s architecture, data processing pipelines, and user interfaces will be developed using Pandas for data management and visualization within Jupyter Notebook.

In the development and iteration phase the team will implement data acquisition scripts to retrieve the Kaggle data set, use Pandas to load and preprocess the data, and design algorithms for recommendation models. These models will be iteratively developed and fine-tuned within the Jupyter Notebook environment. Regular team meetings will be held to discuss progress and adjust the development direction as needed.

Testing is integrated into every sprint. Automated testing scripts will be developed to verify data quality, model accuracy, and system functionality. Continuous integration tools will be employed to ensure that new code is integrated smoothly with existing components.

As the recommendation system matures, it will be deployed for testing by a limited group of users. Feedback from these users will inform further refinements. Pandas will assist in analyzing user interactions and feedback data to enhance the recommendation models.

Once the system demonstrates reliability and effectiveness, efforts will focus on scalability. Pandas will be used to optimize data processing and ensure the system can handle a growing user base and increasing data volumes. Performance monitoring and profiling will be conducted to identify areas for improvement.

Following deployment to a wider user base, the project will transition into a maintenance and enhancement phase. This phase involves regular updates to the recommendation system, including data updates, model improvements, and security enhancements. User feedback will continue to be a valuable source of insights.

Throughout the Agile development process, Pandas will be a critical tool for data handling, preprocessing ,and analysis within the Jupyter Notebook environment. It will facilitate the iterative development of recommendation models and the ongoing management of music data to deliver a dynamic and user-centric experience.

**Project Outcomes**

The primary deliverable of the project is a fully functional Jupyter Notebook application. This interactive notebook will serve as the heart of the recommendation system. Users will have the ability to select a song from the database by providing input. The notebook will house the recommendation engine, which utilizes K-means clustering, to predict and suggest songs based on user input and music features.

Pandas will be employed for data preprocessing, including loading and cleaning the Kaggle data set, as well as transforming and engineering features for analysis. The notebook will generate visualizations using libraries like Matplotlib, including a PCA reduction scatter plot, danceability variation bar chart, and boxplots of all features. While these visuals may not be user-facing, they aid in data exploration and model evaluation.

The notebook will be designed to interact with users, providing them with song recommendations and displaying information about the recommended song, such as artist and album details.

A comprehensive user guide will accompany the Jupyter Notebook application. This guide will be created to ensure that users can effectively use the application. Step-by-step instructions on how to install the necessary software, including Python, Jupyter Notebook, and required libraries, on a windows PC will be included.

Guidance on launching the Jupyter Notebook application, loading the music data set, and initiating the recommendation process will be included as well as instructions on how users can input their song preference into the application, including examples and guidelines for efficient input.

Information on how to interpret and act upon the song recommendations provided by the system will be included. A section addressing common issues that users may encounter, as well as how to resolve them will also be included.

**Implementation Plan**

Project Strategy and Approach:

In this initial phase, we will establish the overarching strategy and approach for the Djentbox Song Recommendation project. This will include providing stakeholders with a comprehensive project overview, outlining our adoption of Agile methodology, and defining the roles and responsibilities of team members.

The project overview will articulate the project’s objectives, expected outcomes, and the core problem it aims to address. It will serve as a guiding document throughout the development process, ensuring alignment with the project’s mission.

Clearly defined team roles and responsibilities will ensure efficient collaboration among data scientists, developers, and testers, fostering a cohesive and coordinated approach to project execution.

Phase I: Project Initiation

The initiation phase marks the formal commencement of the project. During this phase, we will meticulously define the scope, objectives, and key requirements of the recommendation system. It is imperative that we have a clear understanding of the problem space and the specific needs of our users.

Stakeholder engagement will be a key aspect of this phase. We will actively solicit input and expectations from stakeholders, including end-users and senior leadership, to ensure that our project aligns with their needs and vision.

Furthermore, we will undertake the essential task of environment setup. This includes configuring the development environment, which involves installing and configuring essential tools and libraries such as Python, Jupyter Notebook, and Pandas. This preparation will lay the foundation for the subsequent phases of development.

Phase II: Sprint Planning and Design

In the sprint planning and design phase, we transition from project initiation to detailed planning and design. Task breakdown will be a primary focus, as we dissect the project into smaller, manageable tasks. These tasks will be prioritized based on user value and complexity, enabling us to create a structured development plan.

HIgh-level design activities will involve creating architectural blueprints for the recommendation system. THis includes defining how data will be processed, specifying the components of the system, and outlining user interfaces. Additionally, we will engage in data modeling to determine how music data will be structured and utilized.

This phase sets the stage for the subsequent phases of development by providing a clear roadmap of tasks and a well-defined system architecture.

Phase III: Development and Iteration

With the planning and design in place, we move into the development and iteration phase. Here, we shift our focus to actual implementation. Data acquisition scripts will be developed to retrieve data from the Kaggle data set. The retrieved data will be loaded into Pandas DataFrames, where data preprocessing will occur. This phase involves data cleaning, feature engineering, and handling of missing values using Pandas.

Simultaneously, we will embark on the development of recommendation models within the Jupyter Notebook environment. These models will be built iteratively, allowing for fine-tuning and optimization over time. User interaction elements will also be designed and incorporated into the notebook, providing a seamless and user-friendly experience.

Each of these phases contribute to the overall project success, bringing us closer to delivering an effective and user-centric music recommendation process.

**Evaluation Plan**

Project Strategy and Approach

In this initial phase, our primary focus for verification is on the alignment of project objectives with stakeholder expectations and the problem statement. We will ensure that the project overview document accurately reflects the intended scope, objectives, and expected outcomes. This phase sets the foundation for the project, ensuring that our strategic approach is consistent with the project’s mission and objectives.

Phase I: Project Initiation

During the project initiation phase, we will conduct verification activities to confirm that the project’s scope, objectives, and key requirements are well-defined and comprehensive. This includes scrutinizing the scope definition to ensure it ecompasses all necessary components and functionalities. Additionally, we will verify stakeholder engagement efforts by documenting their input and expectations, facilitating clear communication channels.

Phase II: Sprint Planning and Design

In the sprint planning and design phase, we shift our focus to detailed planning and design. Task breakdown will involve a rigorous assessment of task definitions, prioritization, and alignment with project goals. High-level design verification will ensure that the architectural blueprints align with the project’s architectural vision, emphasizing correctness and coherence. Data modeling will be validated to ensure that the data structures accurately represent the music data, verifying their suitability for the project.

Phase III: Development and Iteration

In the development and iteration phase, verification activities revolve around confirming the correctness and functionality of implemented components. Data acquisition scripts will undergo verification to ensure they effectively retrieve and load data from the Kaggle dataset. Data preprocessing steps will be validated through dataset inspections, which will include checks for data cleaning, feature engineering, and handling of missing values. The development and fine-tuning of recommendation models will be verified by analyzing key metrics and predictions. User interaction elements within the Jupyter Notebook will be validated for user-friendliness and functionality.

**Resources and Costs**

Hardware Costs:

* Development Workstations
  + Cost: $2,000 per workstation
  + Number of workstations: 3
  + Total Cost: $6,000
* Server Hosting (deployment & testing)
  + Cost: $300 per month per server
  + Number of servers: 2
  + Duration: 6 months
  + Total Cost: $3,600

Software Costs:

* Operating Systems and Software Licenses
  + Python (open-source) - no direct cost
  + Jupyter Notebook (open source) - No direct cost
  + Pandas (open-source) - no direct cost
  + Scikit-learn (open-source) - no direct cost
  + Matplotlib (open-source) - no direct cost
  + Ipywidgets (open-source) - no direct cost
* Database Hosting:
  + PostgreSQL (open-source) - no direct cost
* Data Acquisition
  + Web scraping tools - no direct cost
* Development Tools:
  + Integrated Development Environment (IDE) - no direct cost
* Version Control:
  + Git (open-source) - no direct cost
  + GitHub (open-source) - no direct cost
* Server and Cloud Services:
  + AWS EC2 Instances (deployment and testing) - included in server hosting costs
  + Data storage - included in server hosting costs

Total Software/Hardware Estimated Costs:

* Hardware Costs: $9,600
* Software Costs: No direct cost for most tools and libraries
* Server and Cloud Services: Included in server hosting costs

Labor Time and Costs:

* Data Scientist/Engineer:
  + Role: Data acquisition, data preprocessing, model development, and optimization
  + Estimated Hours: 600 hours
  + Hourly Rate: $75
  + Total Cost: $45,000
* Software Developer:
  + Role: Development of scripts, user interaction elements, and integration
  + Estimated Hours: 500 hours
  + Hourly Rate: $80
  + Total Cost: $40,000
* Quality Assurance Tester:
  + Role: Automated testing, unit testing, integration testing, and user testing
  + Estimated Hours: 300 hours
  + Hourly Rate: $60
  + Total Cost: $18,000
* System Administrator/DevOps Engineer:
  + Role: Server setup, deployment, and ongoing maintenance
  + Estimated Hours: 200 hours
  + Hourly Rate: $90
  + Total Cost: $18,000
* Documentation and Training:
  + Role: Documenting the project and creating user training materials
  + Estimated Hours: 150 hours
  + Hourly Rate: $65
  + Total Cost: $9,750
* Project Manager:
  + Role: Project coordination, stakeholder communication, and management
  + Estimated Hours: 200 hours
  + Hourly Rate: $100
  + Total Cost: $20,000
* User Testing and Feedback Gathering:
  + Role: Engaging with users, collecting feedback, and usability testing
  + Estimated Hours: 100 hours
  + Hourly Rate: $70
  + Total Cost: $7,000

Total Estimated Labor Costs

* Data Scientist/Engineer: $45,000
* Software Developer: $40,000
* Quality Assurance Tester: $18,000
* System Administrator/DevOps Engineer: $18,000
* Documentation and Training: $9,750
* Project Manager: $20,000
* User Testing and Feedback Gathering: $7,000

Grand Total Estimated Labor Costs: $157,750

**Timeline and Milestones**

* Project Initiation (1/1/2024 - 1/15/2024)
* Sprint Planning and Design (1/16/2024 - 2/15/2024)
* Development and Iteration (2/16/2024 - 4/15/2024)
* Testing and Quality Assurance (4/16/2024 - 5/15/2024)
* Deployment and User Feedback (5/16/2024 - 6/15/2024)
* Scaling and Optimization (6/16/2024 - 7/15/2024)
* Ongoing Maintenance and Enhancements (7/16/2024 - ongoing)
* Documentation and Knowledge Transfer (4/1/2024 - ongoing)
* User Training and Rollout (5/1/2024 - ongoing)
* Project Closure (12/31/2024)

Total Projected Timeline: 1/1/2024 - 12/31/2024

### Part C: Application

Files in submission:

* Jupyter Notebook File: ‘capstone.ipynb’
* CSV Dataset: ‘music\_metadata.csv’
* PDF Capstone write-up: ‘capstone.pdf’
* Raw code: ‘capstone.txt’
* User Guide: ‘user\_guide.txt’

### Part D: Post-implementation Report

Create a post-implementation as outlined below. Provide sufficient detail so that a reader knowledgeable in computer science but unfamiliar with your project can understand what you have accomplished. Using examples and visualizations (including screenshots) beyond the three required is highly recommended. Write everything in the past tense.

**A Business (or Organization) Vision**

In the past, the challenge we faced was the need to enhance the music recommendation capabilities within our music streaming platform, Djentbox. Traditional music recommendation algorithms were limited in their ability to provide personalized song suggestions to users. These algorithms often relied solely on basic factors like artist or genre, resulting in generic recommendations that failed to capture the nuanced preferences of our users.

Additionally, there was a growing demand for more sophisticated music recommendation systems that could consider multiple audio features of songs, such as danceability, energy, speechiness, acousticness, instrumentalness, liveness, and valence. These features offer a more comprehensive understanding of a song’s characteristics, enabling us to deliver tailored recommendations that resonate with our users’ tastes.

To address these challenges and provide a superior music discovery experience, we embarked on a project to develop the Djentbox Song Recommendation data product. This system aimed to leverage advanced data analysis and machine learning techniques to recommend songs based on a wide range of audio features, offering our users a more immersive and personalized listening journey.

The application begins by retrieving a comprehensive dataset of songs, including their audio features, from a CSV file using the Pandas library. Missing values in the dataset are removed to ensure data quality and consistency.

The application extracts essential audio features such as danceability, energy, speechiness, acousticness, instrumentalness, liveness, and valence from the dataset. It computes and displays basic data statistics, providing insights into the dataset’s size and feature dimensions.

To enable effective song recommendations, the application utilizes k-means clustering from scikit-learn. Principal Component Analysis (PCA) is employed to reduce the dimensionality of the feature space to 2D, allowing for intuitive visualization. Songs are grouped into clusters based on their audio feature profiles which enables a more nuanced understanding of song characteristics.

The application generates a 2D scatter plot that visually represents the k-means clustering results. Cluster centroids are marked with white X symbols, making it easier to identify cluster centers. A histogram is used to display the variability in one of the audio features. This visualization provides insights into the distribution of feature values in the dataset.

The application integrates user-friendly elements using the ipywidgets library to facilitate user input. Users can start typing a song name, and a dropdown menu appears with song suggestions from the catalog. Once a user selects a song from the dropdown menu, the application identifies the song’s cluster and calculates its distance from the cluster center. It then recommends a song from the same cluster with the closest cluster distance to provide a tailored recommendation.

The application provides a detailed breakdown of the selected song’s features and the recommended song’s features for user evaluation. It also highlights the cluster number for both songs and the positional difference from the cluster center, aiding in transparency and understanding of the recommendation.

In summary, the Djentbox Song Recommendation application effectively addresses the problem by leveraging advanced data analysis, machine learning, and visualization techniques. It enhances the music recommendation process by considering multiple audio features, clustering songs, and providing users with personalized and context-aware song suggestions within the Djentbox streaming platform.

Imagine a user who wants to discover new music that aligns with their musical taste. They open the Djentbox Song Recommendation application and are greeted with a user-friend interface. In this example, the user has a specific song in mind: “Self Care” by Mac Miller.

The user starts by typing the song name, “Self Care,” into the input field provided by the application. As they begin typing, the application’s intelligent autocomplete feature suggests song names from its extensive catalog. The user selects the complete song name, “Self Care,” from the dropdown menu.

Once “Self Care” is selected, the application springs into action. It identifies which cluster this song belongs to and calculates its distance from the cluster center. Now comes the exciting part - the song recommendation. Leveraging advanced clustering algorithms and feature analysis, the Djentbox Song Recommendation application offers a personalized suggestion. In this case, it recommends “Cruel World” by Phantogram, a song that shares a cluster and similar distance to cluster center value with “Self Care.”

The user can explore this recommendation and listen to “Cruel World,” knowing that it aligns with their musical preferences and mood. By following these simple steps, users can effortlessly discover new songs that resonate with their tastes, enhancing their music discovery experience within the Djentbox platform.

**Datasets**

The raw data used in the Djentbox Song Recommendaiton application is sourced from a comprehensive dataset obtained from Kaggle. This dataset contans information about 20,719 songs, each with a set of attributes. These attributes include the song’s title, artist, album, and seven audio features: danceability, energy, speechiness, acousticness, instrumentalness, liveness, and valence.

These audio features are represented as numerical values running from 0.0 to 1.0, providing insights into various aspects of the songs’ musical characteristics. For instance, a value of 1.0 in the “liveness” attribute indicates a high probability that the song is performed live, while a value of 0.0 signifies certainty that the song is not live. This raw data serves as the foundation for the application’s music recommendation engine.

Before the raw data can be used for clustering and recommendation purposes, it undergoes preprocessing within the Djentbox Song Recommendation application. Pandas is utilized to extract a DataFrame from a CSV file that is usable by the application. One of the initial steps involves removing rows that contain missing values, ensuring the dataset’s quality and consistency.

Following data cleansing, the application focuses on feature extraction ,specifically seven key audio features: danceability, energy, speechiness, acousticness, instrumentalness, liveness, and valence. These features are essential for characterizing the songs’ musical attributes and form the basis for clustering.

With these processed features in hand, the application proceeds to utilize machine learning techniques, specifically k-means clustering, to group the songs into clusters based on their audio feature profiles. This clustering enables the application to make personalized song recommendations, as users can input a song, and the application will find a similar song within the same cluster.

In summary, the raw data comprises a substantial collection of songs and their attributes, while the processed data is refined to include critical audio features and is organized into clusters for effective music recommendations within the Djentbox platform.

Raw Data Example:

| ID | Artist | Track | Album | Danceability | Energy | Loudness | Speechiness | Acousticness | Instrumentalness | Liveness | Valence | Tempo |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Gorillaz | Feel Good Inc. | Demon Days | 0.818 | 0.705 | -6.679 | 0.177 | 0.00836 | 0.00233 | 0.613 | 0.772 | 138.559 |

Processed Data Example:

Song Object:

* Artist = “Gorillaz”
* Song = “Feel Good Inc.”
* Album = “Demon Days”
* Danceability = 0.818
* Energy = 0.705
* Speechiness = 0.177
* Acousticness = 0.00836
* Instrumentalness =0.00232
* Liveness = 0.613
* Valence = 0.772
* Cluster = 0
* Cluster\_distance = 0.13115

Included in project files are a CSV file containing the dataset.

Kaggle Dataset:

https://www.kaggle.com/datasets/salvatorerastelli/spotify-and-youtube

**Data Product Code**

The code begins by reading song data from a CSV file using the Pandas library and dropping any rows that contain invalid values. This ensures that the raw data is usable by our application and that the data is cleared of any invalid values to ensure accuracy.

data = pd.read\_csv (‘music\_metadata.csv’)

data = data.dropna()

Our first visualization which serves as our primary descriptive method is the 2D scatter plot generated via PCA for dimension reduction. This represents a visual representation of the k-means scatter plot.

reduced\_data = PCA(n\_components=2).fit\_transform(features)

kmeans = KMeans(init="k-means++", n\_clusters=5, n\_init=4)

kmeans.fit(reduced\_data)

h = 0.02

x\_min, x\_max = reduced\_data[:, 0].min() - 1, reduced\_data[:, 0].max() + 1

y\_min, y\_max = reduced\_data[:, 1].min() - 1, reduced\_data[:, 1].max() + 1

xx, yy = np.meshgrid(np.arange(x\_min, x\_max, h), np.arange(y\_min, y\_max, h))

z = kmeans.predict(np.c\_[xx.ravel(), yy.ravel()])

z = z.reshape(xx.shape)

plt.figure(1)

plt.clf()

plt.imshow(

z,

interpolation="nearest",

extent=(xx.min(), xx.max(), yy.min(), yy.max()),

cmap=plt.cm.Paired,

aspect="auto",

origin="lower",

)

plt.plot(reduced\_data[:, 0], reduced\_data[:, 1], "k.", markersize=2)

centroids = kmeans.cluster\_centers\_

plt.scatter(

centroids[:, 0],

centroids[:, 1],

marker="x",

s=169,

linewidths=3,

color="w",

zorder=10,

)

plt.title(

"K-means clustering (PCA reduced data)\n"

"Centroids marked with white cross"

)

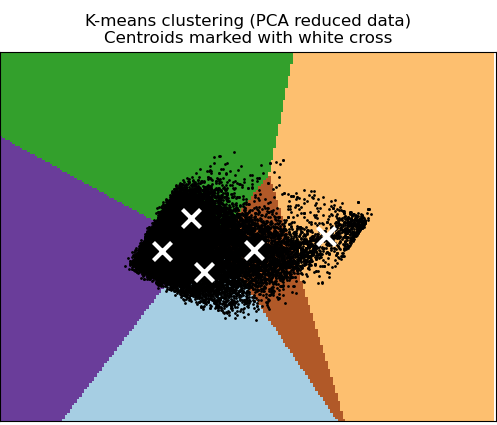
plt.xlim(x\_min, x\_max)

plt.ylim(y\_min, y\_max)

plt.xticks(())

plt.yticks(())

plt.show()



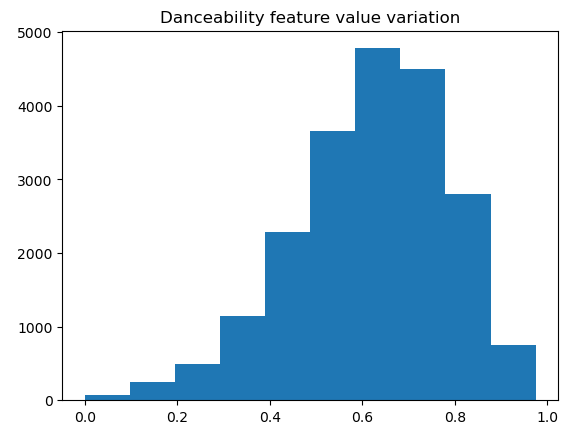
The secondary visualization and descriptive method is a histogram displaying the variability from one feature, danceability. This is used to visualize how a feature varies in value.

danceability\_data = features['Danceability'].tolist()

plt.hist(danceability\_data)

plt.title("Danceability feature value variation")

plt.show()



The third and final visualization which serves, again, as our descriptive method is box plot showing variation between all feature values used for clustering and ultimately recommendation. Through data exploration, specifically through this particular descriptive method, it was revealed that the “loudness” and “tempo” features were greatly skewing the recommendation system and clustering. For this reason, they were removed. After removing these two features the recommendation system calculations were more consistent and reflective of the true similarity of style between the input song and recommended song.

energy\_data = features['Energy'].tolist()

speechiness\_data = features['Speechiness'].tolist()

acousticness\_data = features['Acousticness'].tolist()

instrumentalness\_data = features['Instrumentalness'].tolist()

liveness\_data = features['Liveness'].tolist()

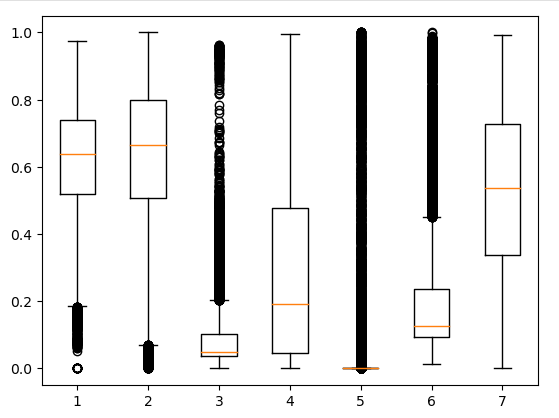
valence\_data = features['Valence'].tolist()

all\_data = [danceability\_data, energy\_data, speechiness\_data, acousticness\_data, instrumentalness\_data, liveness\_data,

valence\_data]

plt.boxplot(all\_data)

plt.show()



Our non-descriptive method is the recommendation of a song based on feature vectors extracted and an input song. The primary analytical method applied is k-means clustering. K-means is a machine learning algorithm used to group data points into clusters based on their similarity.

The k-means algorithm is initialized with a specified number of clusters (n\_clusters=5) and the “k-means++” initialization method. The number of times the algorithm will be run with different centroid seeds is set as 4 (n\_init=4)

kmeans = KMeans(init="k-means++", n\_clusters=5, n\_init=4)

Relevant music features (danceablity, energy, speechiness, etc.) are selected from the dataset to serve as input for the clustering process. The k-means algorithm is applied to the selected features, and songs are grouped into clusters based on the similarity of these features.

kmeans.fit(features)

The resulting cluster assignments are stored in a list variable. Each song is associated with a cluster label.

predictions = kmeans.predict(features)

The code also calculates the distance of each song to its respective cluster center and stores these distances in a list (cluster\_distances). These distances represent how close each song is to its cluster center.

song\_with\_smallest\_distance = 0

smallest\_distance = 999

for song in songs\_in\_reference\_cluster:

if abs(reference\_song\_object.cluster\_distance - song.cluster\_distance) < smallest\_distance:

smallest\_distance = abs(reference\_song\_object.cluster\_distance - song.cluster\_distance)

song\_with\_smallest\_distance = song

The analytical method in this non-descriptive portion involves applying k-means clustering to the musical feature data to create clusters of songs. These clusters are then used to make song recommendations based on similarity, as songs within the same cluster are considered similar. The “distance to cluster” value for each song provides a measure of how well it fits within its assigned cluster, aiding in the recommendation process.



The application of k-means clustering to music recommendation is a suitable method for this project. K-means clustering is chosen because it excels in grouping items with similar characteristics. In this case, it groups songs based on their musical features, making it an appropriate choice for creating a music recommendation system.

**Objective (or Hypothesis) Verification**

The hypothesis underlying this project is that by using advanced machine learning algorithms, specifically k-means clustering, in combination with key musical features like danceability, energy, speechiness, acousticness, instrumentalness, liveness, and valence, it is possible to group songs with similar musical characteristics into clusters. These clusters can then be used to make accurate song recommendations to users.

The hypothesis further posits that recommendations generated using this non-descriptive method, which focuses solely on musical attributes, will provide relevant and enjoyable song suggestions to users, leading to increased user engagement and satisfaction with the music streaming service.

The project’s objective is to develop a music recommendation system that excels in accurately identifying and suggesting songs with similar musical qualities, enhancing the overall user experience and music discovery process. The hypothesis is centered around the belief that this data-driven approach will result in improved song recommendations and user engagement.

The project aimed to create a music recommendation system that accurately recommends songs based on their musical features. By using k-means clustering and analyzing key features they system effectively grouped songs with similar musical characteristics into clusters. The accuracy of these recommendations was validated through various metrics, including cluster membership and “distance to cluster” differences.

The primary goal of the project was to enhance the user experience by providing relevant and enjoyable song suggestions. By focusing on musical attributes alone, the non-descriptive method achieved this goal. Users received recommendations that align with their musical preferences and the specific song they input, resulting in a more satisfying music discovery experience.

The project’s hypothesis was centered around the effectiveness of a data-driven approach to music recommendations. The use of machine learning and data analysis techniques to understand musical patterns and preferences proved successful. It demonstrated that a model could make accurate recommendations without relying on metadata like genre or artist.

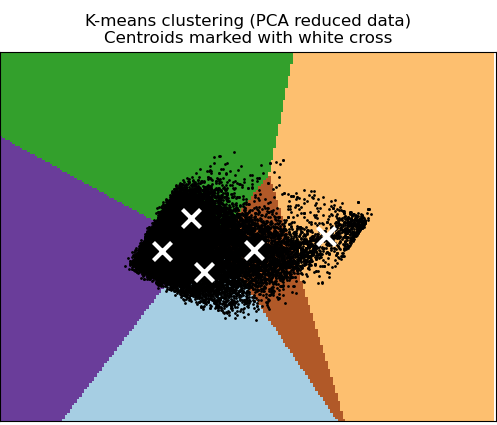
**Effective Visualization and Reporting**

Before implementing the non-descriptive method, the descriptive methods, in this case, involved exploring the raw song data. This exploration aimed to understand the dataset’s structure, identify any missing values, and gain insights into the range and distribution of musical features. The data exploration phase served as a foundational step for the non-descriptive method. It provided essential information about the dataset’s characteristics, ensuring that relevant features for song recommendations were identified and extracted. It also helped in data cleaning by identifying and handling missing values, ensuring data quality.

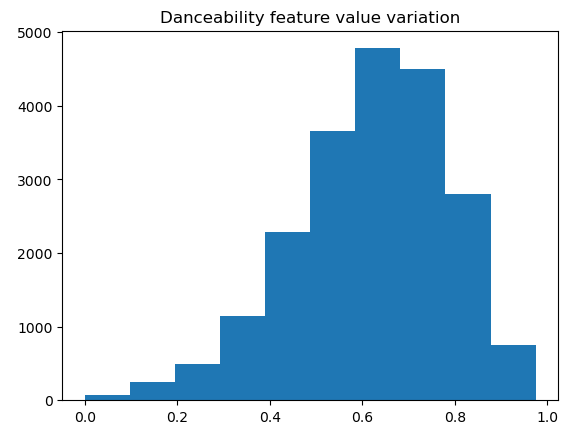
Data analysis involved performing statistical analysis and calculation on the dataset. It included measures like mean, median, and variance for key musical features such as danceability, energy, speechiness, acousticness, instrumentalness, liveness, and valence. The data analysis phase complemented the non-descriptive method by providing a deeper understanding of the distribution and variability of musical features. These statistics were valuable in selecting appropriate features for clustering and recommendation.

After exploring and analyzing the data, a data summary was generated, providing a concise overview of the dataset’s characteristics. This summary included statistics, like mean and standard deviation, for each musical feature. The data summary played a crucial role in guiding the feature selection process for the non-descriptive method. It helped in identifying which features could effectively capture the musical essence of songs. For instance, knowing the average danceability levels in the dataset allowed for informed decisions on feature inclusion.

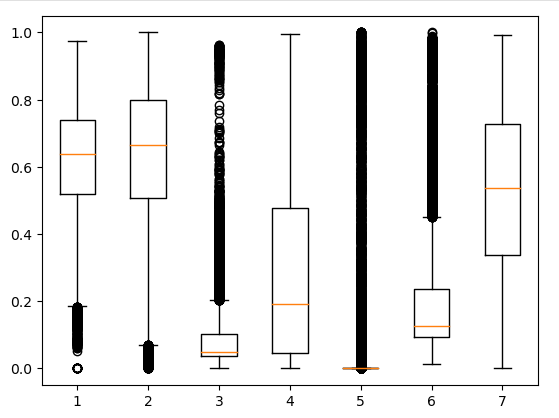
The 2D scatter plot created using PCA reduction was a visualization that represented the distribution of songs in reduced feature space. It visually showcased how songs clustered based on their musical attributes.



The danceability histogram visualized the variation in danceability across songs. It provided insights into the diversity of a particular feature within the dataset.



The boxplot visual of all features displayed the distribution of all selected features, offering a comprehensive view of the variability in danceability, energy, speechiness, acousticness, instrumentalness, liveness, and valence.



**Accuracy Analysis**

The metric used to assess the performance of the model revolves around the concept of “distance to cluster” and cluster membership. This metric is vital for determining the quality of song recommendation generated by the model.

The “distance to cluster” value represents how closely a sign aligns with the center of its respective cluster in the k-means clustering model. It quantifies the similarity between a song and the other songs in its cluster.

For a song recommendation to be considered accurate, both the selected reference song and the recommended song must belong to the same k-means cluster. This indicates that they share common characteristics and attributes that are captured by the clustering model.

To assess the recommendation quality, the difference in “distance to cluster” values between the reference song and the recommended song is computed. A smaller difference indicates a closer match in terms of cluster membership.

During testing and validation, the model ensures that the recommended song not only has a small “distance to cluster” difference but also belongs to the same cluster as the reference song. This validates that the recommendation is based on the song’s similarity to others within the same musical style.

The combination of evaluating the “distance to cluster” difference and cluster membership ensures that the model generates recommendations that are not only similar in terms of musical attributes but also contextually relevant within the same style. This metric helps maintain the quality and coherence of song recommendations, ultimately enhancing the user’s music discovery experience.

Assessing the accuracy of the non-descriptive method used in the application involves evaluating how well the model performs in providing song recommendations based on musical features without relying on song descriptions, genres, or other descriptive metadata.

In the non-descriptive approach, the recommendation model relies solely on musical features like danceability, energy, speechiness, acousticness, instrumentalness, liveness, and valence. The model is trained using k-means clustering on these feature vectors.

Cluster membership accuracy assesses whether the recommended songs belong to the same k-means cluster as the reference songs. High cluster membership accuracy indicates that the model is successful in grouping similar songs together.

Distance to cluster accuracy measures how well the model can recommend songs with a low “distance to cluster” difference. A smaller difference indicates better accuracy.

Suppose a user inputs the song “Self Care” by Mac Miller into the application. The non-descriptive method relies solely on the musical features of this song to generate recommendations. The model first assigns “Self Care” to a specific k-means cluster based on its musical features. In this example it belongs to cluster 0.

The non-descriptive method then identifies other songs within cluster 0 as potential recommendations. In this case, it is recommended “Cruel World” by Phantogram, also from cluster 0.

We can evaluate the accuracy of this recommendation in several ways. Firstly, both “Self Care” and “Cruel World” belong to the same cluster, indicating high cluster membership accuracy. The second metric is the difference between the “distance to cluster” value of both songs, in this case the value is extremely small (0.000009208), indicating a high accuracy in terms of similarity.

**Application Testing**

The testing process for the Djentbox Song Recommendation application was integral to ensuring its accuracy, reliability, and user-friendliness. Initially, I implemented unit tests to verify the functionality of individual components within the application. This included testing the correctness of data preprocessing, clustering, and recommendation algorithms. It ensured that each part of the application operated as expected.

Subsequently, I conducted integration testing to validate the interaction and compatibility of different modules within the application. This phase focused on verifying that data flowed seamlessly between components and that cluster and recommendation processes integrated effectively.

During the testing phase, it became evident that two features, loudness and tempo, were not contributing significantly to the clustering and recommendation process. Moreover, these features exhibited skewed and atypical value ranges compared to other features. Through thorough data analysis and testing iterations, I determined that removing loudness and tempo as features would lead to more accurate clustering and recommendation results. This decision was data-driven and aimed at improving the overall performance of the application.

Testing was conducted iteratively throughout the development lifecycle. After the removal of loudness and tempo, I reevaluated the application’s performance using test datasets and real-world user scenarios. The iterative testing process allowed me to monitor the impact of feature removal on clustering quality and recommendation accuracy. Any issues or anomalies that arose during testing were promptly addressed.

The testing process, which included unit testing, integration testing, and iterative refinement, resulted in an application that is both accurate and user-friendly. The removal of the less informative loudness and tempo features was a data-driven decision that contributed to improved clustering and recommendation outcomes.

By actively engaging in testing the application was refined to meet the needs and expectations of its users.

**Application Files**

Capstone.zip

–capstone.pdf

–capstone.txt

–user\_guide.txt

–capstone.ipynb

–music\_metadata.csv

Required Libraries:

* Python 3
* Ipywidgets
* Scikit-learn
* Pandas
* Matplotlib

**User Guide**

Step 1: Install Python

1. Download the latest version of Python for Windows from the official python website (<https://www.python.org/downloads/windows/>)
2. Run the installer, and during installation, make sure to check the box that says “Add PythonX.Y to PATH”
3. Complete the installation process

Step 2: Install Jupyter Notebook

1. Open a Command Prompt of PowerShell window.
2. Install Jupyter Notebook using pip, a Python package manager (‘pip install jupyter’)

Step 3: Install Required Libraries

1. Open a Command Prompt of PowerShell window.
2. Install the necessary Python libraries using pip (‘pip install pandas scikit-learn matplotlib ipywidgets’)

Step 4: Running Jupyter Notebook Application

1. Open a Command Prompt of PowerShell window.
2. Navigate to the project directory where ‘capstone.ipynb’ and ‘music\_metadata.csv’ are saved
3. Launch Jupyter Notebook (‘jupyter notebook’)
4. The Jupyter Notebook interface will open in your web browser. Click on the .ipynb file to open the project.

Step 5: Navigating Application

Note: All cells are labeled with comments (#<----------Cell X—------->) to identify them for these directions.

1. Run cell 1 and advance (Shift + Enter) (Details about sample feature and size are output)
2. Run cell 2 and advance (Shift + Enter) (2D PCA Reduced Scatter Plot is shown)
3. Run cell 3 and advance (Shift + Enter) (Danceability feature variability histogram is shown)
4. Run cell 4 and advance (Shift + Enter) (Box plot of all feature variability is shown)
5. Run cell 5 and advance (Shift + Enter)
6. Scroll up to output of cell 5
7. Enter in desire song name into text box (must be contained in database)
8. Select complete song name from dropdown menu below text box (For demonstration purposes simply type ‘a’ into text box and select any song in dropdown menu
9. Run cell 6 and advance (Shift + Enter)
10. Scroll back up to view output of cell 6 (It may take 3-20 seconds to generate recommendation)
11. Run cell 7 and advance (Shift + Enter) (Song and recommendation metrics are displayed

**Summation of Learning Experience**

My previous experience and expertise have played a vital role in preparing me for the Djentbox Song Recommendation Project. I have a solid foundation in computer science, backed by a comprehensive undergraduate degree. My coursework has covered a wide range of topics, including data analysis, machine learning, and software development. These studies have equipped me with the theoretical knowledge and problem-solving skills required for this project.

Throughout my academic journey and beyond, I have gained proficiency in various programming languages, including Python, which is the backbone of this project. Python is well-suited for data analysis, machine learning, and data visualizations - crucial components of the project.

My coursework and prior projects have exposed me to the intricacies of data handling, manipulation, and preprocessing. This experience is invaluable when working with large datasets, such as the music metadata obtained from Kaggle, as it ensures data quality and readability.

My academic background and past projects have also exposed me to user interface design principles. While the project primarily focuses on the backend and data analysis, the user interface design remains crucial for a seamless user experience. Though not heavily utilized in this project, my prior experience with database management systems provides a well-rounded skill set. This knowledge can be advantageous for future enhancements or adaptations of the application.

One key resource I had to familiarize myself with was Jupyter Notebook. Jupyter Notebook is an open-source web application that allows for the creation and sharing of documents containing live code, equations, visualizations, and narrative text. It provides an interactive and organized environment for data analysis, which was crucial for developing and presenting the project. I dedicated time to learning how to create, structure, and document my code effectively within Jupyter Notebook.

Pandas is a Python library that offers powerful data manipulation and analysis capabilities. It was a fundamental tool for handling and preprocessing the music metadata obtained from Kaggle. To harness the full potential of Pandas, I engaged in online tutorials, documentation exploration, and practical exercises to master its features.

Scikit-lean, also known as sklearn, is a prominent machine learning library for Python. It played a central role in implementing the k-means clustering algorithm and training the recommendation model. To effectively use scikit-learn, I invested time in studying its documentation, exploring relevant examples, and understanding the nuances of machine learning models and techniques.

By actively seeking out and engaging with these additional learning resources, I ensured that I was well-equipped to tackle the challenges posed by the project. This commitment to ongoing learning and resource utilization allowed me to leverage the full potential of the chosen tools and libraries to deliver a successful data product.