Fundamental of Big Data Analytics (DS2004)

Project

Deadline: 12-05-2024

Course Team		
Dr. Muhammad Ishtiaq	Course Coordinator	m.ishtiaq@nu.edu.pk
Ms. Kainat Iqbal	Course Instructor	kainat.iqbal@isb.nu.edu.pk
Ibrahim Bin Umair	Teaching Assisant	i200567@nu.edu.pk
Muhammad Huzaifa Khan	Teaching Assisant	i212689@nu.edu.pk
Hashim Muhammad Nadeem	Teaching Assisant	i211675@nu.edu.pk

Project Guidelines:

- This project is meant to be completed as a team.
- Only **one team member** should submit the project. Multiple or duplicate submissions will lead to a deduction of marks.
- To submit the project, please create a GitHub repository and upload your source codes along with your result file. **Ensure the repository is public.** Share **only** the repository link on Google Classroom for submission. Do *not* include any additional materials on Google Classroom, as marks will be deducted for doing so.
- Additionally, you must provide a comprehensive report detailing your work and findings. This report should be written within the **README.md** file located in your GitHub repository **(Example)**. Avoid submitting any separate document files.
- You have the flexibility to utilise any programming language that is compatible with Apache Hadoop, such as Python, Java, or C/C++.
- You may refer to online sources such as websites and/or ChatGPT, but make you can explain the code you have written.
- To earn bonus marks, please use comments and adhere to correct **PEP 8** coding conventions.

Plagiarism Policy:

Plagiarism is a grave academic offense that can severely undermine your academic integrity and reputation. Any instance of a student found to have plagiarised their assignment, whether from a peer or external source, will be subject to strict consequences. This may result in a zero score for the current or all assignments, or in severe cases, even a failure in the course. Furthermore, all instances of plagiarism will be promptly reported to the Disciplinary Committee for further action.

Create Your Own Spotify Experience

[150 Marks]

Spotify is a digital music streaming service that provides access to millions of songs, podcasts, and videos from artists all around the world. Users can listen to music for free with advertisements or subscribe to a premium plan for an ad-free experience and additional features like offline listening and high-quality audio.

Spotify's music recommendation system is powered by machine learning algorithms that analyse user behaviour, preferences, and listening habits to generate personalised recommendations. These algorithms consider factors such as the songs users listen to, how often they listen, playlists they create, and artists they follow.

Additionally, Spotify utilises collaborative filtering techniques, content-based filtering, and natural language processing to enhance recommendation accuracy. The system continuously learns and adapts to user feedback, ensuring that recommendations stay relevant and diverse over time.

Your final assignment is to develop a streamlined alternative to Spotify. This project will feature a **music recommendation system**, **playback**, and **streaming capabilities**, alongside **real-time suggestions** derived from user activity.

Phase #1: Extract, Transform, Load (ETL) Pipeline [40 Marks]

- The first task involves creating an Extract, Transform, Load (ETL) pipeline utilising **Free Music Archive (FMA)**, a readily available dataset ideal for assessing various endeavours in music information retrieval (MIR), a domain focused on navigating, querying, and structuring extensive music libraries.

- You'll be working with the `fma_large.zip` dataset, comprising 106,574 tracks, each lasting 30 seconds, and spanning 161 unevenly distributed genres. Compressed, the dataset totals **93 GiB in size**. Moreover, you may find the `fma_metadata.zip` data necessary for track details like title, artist, genres, tags, and play counts, covering all 106,574 tracks. Your selection of features will vary based on your approach to music recommendation.

- After downloading, you'll use Python to load the dataset and execute important feature extraction methods like Mel-Frequency Cepstral Coefficients (MFCC), spectral centroid, or zero-crossing rate. These techniques will convert the audio files into numerical and vector formats. Additionally, consider exploring normalisation or standardisation techniques, as well as dimensionality reduction, if necessary, as they can greatly enhance the accuracy of your recommendation model.

- Finally, due to the dataset's vast size, it's important to store it in a scalable and accessible manner. Fortunately, MongoDB fulfils these requirements seamlessly. After transformation, you can effortlessly load the audio features into a MongoDB collection for further utilisation.

Phase #2: Music Recommendation Model [70 Marks]

- Now that the data is securely stored in MongoDB, the next step involves using Apache Spark to train a music recommendation model. You have the option to leverage Apache Spark's MLlib machine learning library or explore deep learning methodologies for enhanced accuracy, utilising emerging frameworks like the TorchDistributor API for PyTorch. Algorithms such as collaborative filtering and Approximate Nearest Neighbours (ANN) can be used in this process.

- Following the training phase, it is important to assess your music recommendation model using different evaluation metrics. It's worth noting that hyperparameter tuning for the model holds considerable importance, and the parameters you select must be supported by your implementation.

Phase #3: Deployment [30 Marks]

- Upon completing the model training, your next task is to deploy it onto a web application. However, the challenge lies in the fact that it's not just any web application but a streaming service. Your objective is to develop an interactive music streaming web application that incorporates the mentioned features. Crafting a well-structured and user-friendly web interface carries substantial weightage. You have the freedom to utilise frameworks such as Flask or Django for this purpose.

- You'll leverage Apache Kafka to dynamically generate music recommendations for users in real-time, using historical playback data to inform future suggestions. All of this will seamlessly occur concurrently in the background of the web application.

- The web application must *not* include a form prompting users to upload audio files. Instead, recommendations must be exclusively generated by Apache Kafka in real-time, monitoring user activity to tailor suggestions accordingly.

Phase #4: Reporting [10 Marks]

- Finally, you are required to compose a comprehensive report detailing your methodology and findings. This report should contain the results of your music recommendation system, evaluating its effectiveness and implementation. Your findings will be documented within the `README.md` file in your GitHub repository.

In this project, we suggest **exploring additional cutting-edge big data technologies** such as Apache Arrow and Microsoft Azure Cloud. You might find it beneficial to incorporate these elements if they align with your specific requirements.

Break a leg! :D