A logo for college computing

Description automatically generated

**Assessment Cover Page**

|  |  |
| --- | --- |
| *Student Full Name: Cristhian Elson Pereira Macedo* |  |
| *Student Number: 2024104* |  |
| *Module Title: Data Visualization Techniques - Machine Learning for Business* |  |
| *Assessment Title: CA2* |  |
| *Assessment Due Date: 24th November 2024 23:59* |  |
| *Date of Submission: 24th November 2024* |  |

**Declaration**

By submitting this assessment, I confirm that I have read the CCT policy on academic misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source.

I declare it to be my own work and that all material from third parties has been appropriately referenced.

I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution.



**Anime Recommendation**

**Integrated CA2**

**Data Visualization Techniques and Machine Learning for Business**

**Cristhian Elson Pereira Macedo**

**2024104**

Higher Diploma in Science in Data Analytics for Business

Data Visualization Techniques (DVTDA) and Machine Learning for Business (MLBDA)

Lecturer: David McQuaid and Dr. Muhammad Iqbal

CCT College Dublin

Dublin, Ireland

2024



**SUMMARY**

[Table of Figures iv](#_Toc183383553)

[1. Introduction 5](#_Toc183383554)

[2. Data Understanding 5](#_Toc183383555)

[3. DATA PREPARATION 5](#_Toc183383556)

[4. Exploratory data analysis 6](#_Toc183383557)

[5. MACHINE LEARNING 7](#_Toc183383558)

[6. Interactive Dashboard 13](#_Toc183383559)

[INFORMATION 17](#_Toc183383560)

[DATA DICTIONARY 19](#_Toc183383561)

[References 21](#_Toc183383562)

# **Table of Figures**

[Figure 1 Recommender System with Content-Based Result 8](#_Toc183383543)

[Figure 2 Recommender System with Content-Based Result 2 8](#_Toc183383544)

[Figure 3 Collaborative Filtering Items to be Recommended 8](#_Toc183383545)

[Figure 4 Collaborative Filtering - Item to Item 9](#_Toc183383546)

[Figure 5 Collaborative Filtering - User to User 9](#_Toc183383547)

[Figure 6 Apriori Frequent Item sets 10](#_Toc183383548)

[Figure 7 Apriori support, confidence and lift results 11](#_Toc183383549)

[Figure 8 FP Growth Frequent Item sets 12](#_Toc183383550)

[Figure 9 FP Growth support, confidence and lift results 13](#_Toc183383551)

[Figure 10 Animes Data Analysis Dashboard 15](#_Toc183383552)

# **Introduction**

This project aims to complete the second Assessment Task Integrated for the module Data Visualization Techniques and Machine Learning for Business from the course Higher Diploma in Science in Data Analytics for Business by CCT College Dublin, this is a document to describe the project itself whose name is "CristhianMacedo\_Integrated\_MLB\_DVizHDip\_CA2" and should use this document for more clarification. This Data Visualization Techniques and Machine Learning for Business project uses the programming language Python, the environment of Anaconda Navigator with Jupyter Notebook, with CRISP-DM methodology as project management.

The area that has been chosen to be covered the Recommendation System subject is Animes containing four databases called “animalist, rating\_complete, anime and anime\_with\_synopsis” Those data sets contain information about 17562 animes with 325.772 different users collected from the website myanimelist.net database available on Kaggle website with 10 points of Usability.

# **Data Understanding**

Starting the Project importing Pandas library and warnings, but once need to import new libraries and technologies, it will add in this topic, the technologies used in this project: “pandas, numpy, matplotlib, seaborn, missingno ,plotly ,time ,dash\_bootstrap\_components ,tkinter ,sklearn ,googletrans ,fuzzywuzzy, scipy, copy, mlxtend, dash and warnings”.

Load the data frame whose name is “anime” in approximately 0.07 seconds, created a code to be possible to see all columns, next it checked the head of it, shape where it returns 17562 rows (observations) with 35 columns (features), next, checking the info of the data, memory usage, a quick view in the “describe” method to check some statistical information, analysed null values, where in this case, there are none to be dealt with, it was analysed duplicate rows and returned none duplicates, checked the data type of features where it was possible to notice, that a few features should be numeric are presenting type as "object".

# **DATA PREPARATION**

After completing the data understanding phase, it noticed a few things to be corrected and updated, first, it understood the unique values of the data frame, where was found 'Unknown' values of diverse features, also in the feature Aired has some years with the description "to ?" in the end, next with a “for loop” technique it was possible to check within features had Unknown as value, next created a function to keep the graphics in the same pattern and plot the difference of real values and Unknown values for each feature.

Next, it was dealt with these unknown values; Score feature: used the “SimpleImputer” technique to input values as strategy median parameter; Gender feature: deleted the 0.36% of it, English Name feature: deleted the whole feature once it already has one feature with the anime's title, Japanese Name: used a google translate library to translate it from English to Japanese, Type feature: deleted the 0.21% of it, Episodes feature: Inputed values, Object Features: Aired, Premiered, Producers, Licensors, Studios, Source, Duration, Rating and Ranked opted to keep these features with unknown values, Features: Ranked, Score-10, Score-9, Score-8, Score-7, Score-6, Score-5, Score-4, Score-3, Score-2 and Score-1: Inputed values.

After dealing with the features, all of them got the corrected data type, next renamed some features to maintain a standard and simple manipulation when needed, and prepared the Genres feature for Machine Learning, some of the values had space and hyphen symbol separating the words, opted to add an (underscore symbol "\_") to separate it, next, got 43 different genre names in total, The unique genres are: Action, Adventure, Comedy, Drama, Sci\_Fi, Space, Mystery, Shounen, Police, Supernatural, Magic, Fantasy, Sports, Josei, Romance, Slice\_ofLife, Cars, Seinen, Horror, Psychological, Thriller, Super\_Power, Martial\_Arts School, Ecchi, Vampire, Military, Historical, Dementia, Mecha, Demons, Samurai, Game, Shoujo, Harem, Music, Shoujo\_Ai, Shounen\_Ai, Kids, Hentai, Parody, Yuri and Yaoi, next plotted it in a graphic.

Also created a new data frame called “anime\_genres” after using the method “Dummies” to separate each Genre in a feature and plotted it in a graphic next, and finally created a new column in the Feature Creation topic with the sum of all Score features.

# **Exploratory data analysis**

After completing the data understanding phase, it was started an exploratory data analysis step even though the data set was already ready to use, where were checked the distribution of each numerical feature using a "distplot" and "kdeplot" returning their respective distribution and density, also how skewed these features are to the left or right, or in case of a normal distribution, and points of mean and median in the distribution also with their Standard Deviation value, created also a “hist plot” and “boxplot” to visualise in another perspective.

Next, it created a "countplot" to visualise all categorical features, it opted to remove the description of all features once there were too many values to show and turned the result into a big mess, in this case, the main goal is to show how the values are in the categorical features.

Next, different graphs with several columns were created to analyse and better understand how the data information is distributed and perform analyses receiving simple answers, such as what are the top 10 genres, which is the largest studio that makes anime, or which is the best producers, best score, number of people who watched the anime, who are watching, who are on pause, who gave up or are planning to watch, and so on, the graphics that were used are: bar plot, treemap, histogram, pie and box plot.

# **MACHINE LEARNING**

After completing the exploratory data analysis phase, it started Machine Learning with two topics, the first topic is about creating a Recommendation System using “Content and Collaborative filtering” using user-user and item-item in collaborative filtering and the second topic is about Market Basket Analysis using “Apriori and FP growth” algorithms.

The Recommendation System is useful for companies and customers, as it helps to make decisions on big options that are sometimes not easy and turn into a waste of time, for example, for customers it is simple to receive recommendations based on their history, as we already know what they like and we can offer similar content/products/items based on their tastes, for companies, they could do a customer study to be carried out likes, sales, popularity, usage, patterns, preferences, actions, tastes analysed and so on, with the results, it will be possible to offer more of this content/products/items and similar ones, also never run out of stock if it is a tangible product or item turning this beneficial for both parties.

First, creating a recommendation using the Content-Based Recommender System, in this case, the Genre column will be used, but the following columns could be used as well: Type, Producers, Licensors, Studios, Source and Rating; next it was created a Vectorizer Object to remove/separate English stop words/words, in this case, working well once the genres are already ready to use, next it will compute a similarity score to check the similarity between the genres, in this case, instead of using "cosine\_similarities()" to calculate, it will use "linear\_kernel()" based on "TF-IDF scores", using linear\_kernel keep the compute faster, also could be used others similarity metrics such as "manhattan, euclidean, Pearson, cosine similarity scores" and so on.

Next, create the functions of the engine and finally the function of content-based recommender as an example below:

A screenshot of a computer

Description automatically generated

Figure Recommender System with Content-Based Result

Next, it was created an example of a Recommender System using "CountVectorizer" instead of "TfidfVectorizer" to vectorise Objects and remove/separate English stop words/words, after created the matrix and so on, got the result below:

A screenshot of a computer

Description automatically generated

Figure Recommender System with Content-Based Result 2

Next, it created the Collaborative Filtering for this scenario, it will be necessary to import the second data frame, which has "user\_id, anime\_id and rating features", to load the entire database with 57633278 observations, it takes approximately 4.23 seconds, but will be used just 50000 observations from the whole database, which takes approximately 0.01 seconds to import, it opted to use just part of the data frame once the original data with "57633278" is too large and the algorithm takes a long time to run, and every time it was tested the program always stopped working and crash.

After analyses, created the matrix, functions and so on using “csr\_matrix” and “NearestNeighbors” it were got the follow results:

A number on a white background

Description automatically generated

Figure Collaborative Filtering Items to be Recommended

Next, created a Collaborative Filtering - Item to Item and got the following results with their respective predicted ratings.

A number on a white background

Description automatically generated

Figure Collaborative Filtering - Item to Item

Next, created a Collaborative Filtering - User to User, but for this scenario, it will be necessary to import the second data frame, which has "user\_id, anime\_id and rating features". It will be named "anime\_rating" after loading the data set "rating\_complete.csv" from the folder "animes\_data" using the command ".read\_csv()", also used the method "time" to get an idea of ​​how long it takes to import the database, also created a new data frame called "Mean" to save the "mean" value of the ratings, next merged in a new data frame called "Rating\_avg" and finally created a new featured called "adg\_rating", after analysis got the follow results:

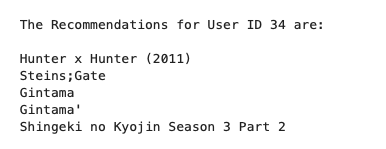


Figure Collaborative Filtering - User to User

After using, "Content-Based Recommender System and Collaborative Filtering" it was noted that each of them has a form of recommendations, and the input form also varies based on the database, "Content-Based" was good for recommending items based on genres, titles, studios, producers, user's profile, etc., which already exist in the database without any activities of the user, "Collaborative" on the other hand, is different, it recommends based on user activities, user behaviour, in this case, based on their ratings can be recommended products, or also it can be recommended base on activities from user to user.

Also, the recommendations for the actual scenario were partially favourable where the databases have a lot of information, such as user IDs, anime IDs, user ratings, and so on. that is essential for some analyses using Content-Based and Collaborative Filtering, this dataset made it possible to explore and create various analyses, unfortunately, the time is short to explore it dipper, but a few analysis was carried out as far as possible, although, the variety for this database is immense and can be updated and explored.

In the second topic is about Market Basket Analysis using “Apriori and FP growth” algorithms, for this scenario, it will be necessary to import the third data frame, which has "user\_id, anime\_id, rating, watching\_status and watched\_episodes features", to load the entire database with 109224747 observations, it takes approximately 11.61 seconds but will be used just 50000 observations from the whole database, which takes approximately 0.02 seconds to import.

After checked the values, create the filters, enconded the values "0 to False and 1 to True", once the "apriori()" method does not work with numerical values, just Booleans, applied apriori generating the frequent items using "min\_support" as 0.06 it was tested with other values, and it opted to keep it once the results were good enough, next, created two "for loops" to print all values into "itemsets" values, and next the second print the ID an Anime Title, results below:

A screenshot of a computer

Description automatically generated

Figure Apriori Frequent Item sets

Next, generating the rules using "association\_rules" with their corresponding support, confidence and lift, results below:

A screenshot of a computer

Description automatically generated

Figure Apriori support, confidence and lift results

Next, similar to Apriori but using FP Growth Algorithm, got the results below:

A screenshot of a computer

Description automatically generated

Figure FP Growth Frequent Item sets

Next, generating the rules using "association\_rules" with their corresponding support, confidence and lift.

A screenshot of a computer

Description automatically generated

Figure FP Growth support, confidence and lift results

For the test applied using Market Basket Analysis with "Apriori and FP growth" algorithms, there did not appear to be any visible divergences using the two methods. There is possibly significance in computational performance issues. Both methods are similar, except for the word "apriori being changed to fpgrowth". Both are quite similar, but the results using "support, confidence and lift" between apriori and fpgrowth are different, sometimes they appear to be the same, but they are significantly different from each other.

# **Interactive Dashboard**

An attempt using Panel Dashboard was created but was deleted because it was not possible to go very deep with this library then next, it was used Tkinter to create a visualisation with Windows, but it would be very manual and require a lot of time to create a decent visualisation. Also, the visualisation is not becoming very interesting, so the attempt to use Tkinter was kept for visualisation and future upgrades with improvements to create an acceptable dashboard using this library.

Next, it was created a simple Dash Plotly to get used to it, after read about Interactive Dashboard using Dash Plotly in the articles "Dash in 20 Minutes" (Plotly, 2024a) and "Dash Layout" (Plotly, 2024b), inspired by the Explore Dash examples in "Data Visualization & Dashboards" (Plotly, 2024c).

Finally, inspired by the Explore Dash examples in "Data Visualization & Dashboards" (Plotly, 2024c) example of "IMDb Analysis for Movies & Series - Visualize and analyze IMDB data for top movies and series and get recommendations based on your taste." by "Information Technology Institute (ITI)" (Mohamed Elsayed, 2024), it was created a decent Interactive Dashboard.

First, installed the "dash\_bootstrap\_components" dependencies using pip install, created a function called "generate\_visualizations" to be responsible for plotting 4 graphics, and created the main Interactive Dash code was built following these steps:

* Step #1: Preparing to build the Dash app / Initialize the app
* Step #2: Building the layout of the dashboard
  + Building the layout Style
  + dropdown options to recommend
  + Getting some values to present it
  + Building the layout of the App
* Step #3: Adding interactivity to the dashboard
  + Function to get recommendations
  + Callback to update image container based on dropdown selection
  + tabs-content
* Step #4: Running the dashboard

The final Data Analysis Dashboard it was created and visualised as “*Figure 10 Animes Data Analysis Dashboard*” below:

A screenshot of a computer

Description automatically generated

Figure Animes Data Analysis Dashboard

It was not easy and simple to create this visualisation, first, it was necessary to understand how the organisation of codes works using Plotly Dash. Thanks the previous example helped to clarify that there are four steps to be followed:

* Step #1: Preparing to build the Dash app
* Step #2: Building the layout of the dashboard
* Step #3: Adding interactivity to the dashboard
* Step #4: Running the dashboard

Based on these steps, the panel was created for younger adults (18 - 35 years) old, important features were chosen to compose this layout as a main overview to present to the user, showing the number of animes, genres, members and studios composed in this base, also graphics such as Total of Score per Type of Animes, Top 10 Genders between Animes, Top 5 Producers by Animes and Popularity Distribution was important to complete the overview of it.

Instead of creating external .py files (python extension), it was decided to leave the functions internally, for example, the "generate\_visualizations" function could be created in an external file and imported, but it was decided to keep everything within the same document to keep it centralised, with only the images that are in an external folder being imported into the document.

An option for recommendations within the visualisation was also created if it is of interest to the user. The buttons are limited to an overview of all types of anime within the database, but new improvements can be created to show different aspects of the database and present them to the public.

The dataset is suitable for Machine Learning models in an online retail business, once there are important features like Anime ID, User ID, Score, Ratings, Genders, Type, Producers, Studios, Popularity and so on, with this information, it is possible to create recommendations using content-based recommender and collaborative filtering user-user and item-item, also for the use of Market Basket Analysis using Apriori and FP Growth Algorithm and a huge range of graphics visualisation.

Regarding visualisations using Plotly, a standard was created to keep all graphs created in the same visualisation standard, using font size 16 for titles, font family "Arial", font colour in "black", positions and proportions in the same alignment and green colouring.

Opted to use a green range of colour once, "In color psychology, colors made up of long wavelengths are considered "arousing or warm," whereas colors such as green that have shorter wavelengths are "relaxing or cool." Whereas our eyes must adjust to see colors with longer wavelengths, they don't need to adjust at all to see cool colors." (Cherry, 2023), also "Green Is Calming, Green Is Healthful, Green Is Natural, Green Is Optimistic" (Cherry, 2023).

The dashboard was created part by part, creating each element, testing and saving. When an error was found, it looked for the solution, reading the code and understanding more about the block codes to get solutions. Based on the creation block by block, all the final code was created. For example, first, the main layout block was created, then the icons and buttons by rows and columns, testing colours, size, etc. After the visualisation was done, the values ​​were created to be put internally as results.

Then, the dropdown for recommendations was created. However, the recommendations were quite challenging, so the graphs were created first and then tested one by one. After the graphs were finished, it focused on the recommendations, where the functions were tested outside the main code. The main problem was that the "dcc.Link" function, which was expecting link data, but since the database does not have this information, it was commented out as nothing, and so everything worked correctly.

However, updates can be made to the design and upgrades to improve the dashboard. Unfortunately, there is not enough time to test Streamlit. So the panel was created using Plotly Dash with the focus of showing people a little more about anime and its popularity, the number of members who follow and watch anime, also the number of active studios creating anime, and, last but not least, recommendations which would be the main point for a new and old user, and the whole community.

# **INFORMATION**

**Google Drive with Data sets by Cristhan Macedo on CCT E-mail**

https://drive.google.com/drive/folders/13YLds\_zNIfJyyT6EeDVrUzXZgO09kDjW?usp=sharing

Recommendation data from 320.0000 users and 16.000 animes at myanimelist.net

This dataset contains information about 17.562 anime and the preference from 325.772 different users. In particular, this dataset contain:

* The anime list per user. Include dropped, complete, plan to watch, currently watching and on hold.
* Ratings given by users to the animes that they has watched completely.
* Information about the anime like genre, stats, studio, etc.
* HTML with anime information to do data scrapping. These files contain information such as reviews, synopsis, information about the staff, anime statistics, genre, etc. Also, the code used to collect the data is available at github: https://github.com/Hernan4444/MyAnimeList-Database.

**Warning: this dataset includes information about anime for adults (hentai).**

The data was scrapped between February 26th and March 20th.

* The "html" folder contain 1 zip per anime (17.562 different anime). Each zip contains different HTML pages scrapped from MyAnimeList. The scrapped pages are:

1. Main page
2. Reviews
3. Recommendations
4. Stats
5. Characters & Staff

I uploaded 2 files as example to don't increase the size of this dataset. All HTML files are in this link: https://drive.google.com/drive/folders/12ghJk-sWyXXORoLBUpPirK4YdtIaZPV\_?usp=sharing

**Acknowledgements**

Thanks to:

1. MyAnimeList for providing anime data.
2. Jikan API for provide users preference.
3. Pontificia Universidad Católica de Chile for provide servers to run the code.

**Inspiration**

1. Have an HTML files to experience the scraping exercise without the delay of each requests.
2. Experiment with different types of recommended. For instance, collaborative filtering or based on context like stats, genre, seiyus, reviews, synopsis, etc.
3. Use this information to build a better anime recommended system.
4. Identifying which feature allows us to build the best anime recommended system.

Ideas to the future

Build the same dataset with manga and novel.

# **DATA DICTIONARY**

"animelist.csv" have the list of all animes register by the user with the respective score, watching status and numbers of episodes watched. This dataset contains 109 Million row, 17.562 different animes and 325.772 different users. The file have the following columns:

* user\_id: non identifiable randomly generated user id.
* anime\_id: MyAnimeList ID of the anime. (e.g. 1).
* score: score between 1 to 10 given by the user. 0 if the user didn't assign a score. (e.g. 10)
* watching\_status: state ID from this anime in the anime list of this user. (e.g. 2)
* watched\_episodes: numbers of episodes watched by the user. (e.g. 24)
* "watching\_status.csv" describe every possible status of the column: "watching\_status" in animelist.csv.

"rating\_complete.csv" is a subset of "animelist.csv". This dataset only considers animes that the user has watched completely (watching\_status==2) and gave it a score (score!=0). This dataset contains 57 Million ratings applied to 16.872 animes by 310.059 users. This file have the following columns:

* user\_id: non identifiable randomly generated user id.
* anime\_id: - MyAnimelist ID of the anime that this user has rated.
* rating: rating that this user has assigned.

"anime.csv" contain general information of every anime (17.562 different anime) like genre, stats, studio, etc. This file have the following columns:

* MAL\_ID: MyAnimelist ID of the anime. (e.g. 1)
* Name: full name of the anime. (e.g. Cowboy Bebop)
* Score: average score of the anime given from all users in MyAnimelist database. (e.g. 8.78)
* Genres: comma separated list of genres for this anime. (e.g. Action, Adventure, Comedy, Drama, Sci-Fi, Space)
* English name: full name in english of the anime. (e.g. Cowboy Bebop)
* Japanese name: full name in japanses of the anime. (e.g. カウボーイビバップ)
* Type: TV, movie, OVA, etc. (e.g. TV)
* Episodes': number of chapters. (e.g. 26)
* Aired: broadcast date. (e.g. Apr 3, 1998 to Apr 24, 1999)
* Premiered: season premiere. (e.g. Spring 1998)
* Producers: comma separated list of produducers (e.g. Bandai Visual)
* Licensors: comma separated list of licensors (e.g. Funimation, Bandai Entertainment)
* Studios: comma separated list of studios (e.g. Sunrise)
* Source: Manga, Light novel, Book, etc. (e.g Original)
* Duration: duration of the anime per episode (e.g 24 min. per ep.)
* Rating: age rate (e.g. R - 17+ (violence & profanity))
* Ranked: position based in the score. (e.g 28)
* Popularity: position based in the the number of users who have added the anime to their list. (e.g 39)
* Members: number of community members that are in this anime's "group". (e.g. 1251960)
* Favorites: number of users who have the anime as "favorites". (e.g. 61,971)
* Watching: number of users who are watching the anime. (e.g. 105808)
* Completed: number of users who have complete the anime. (e.g. 718161)
* On-Hold: number of users who have the anime on Hold. (e.g. 71513)
* Dropped: number of users who have dropped the anime. (e.g. 26678)
* Plan to Watch': number of users who plan to watch the anime. (e.g. 329800)
* Score-10': number of users who scored 10. (e.g. 229170)
* Score-9': number of users who scored 9. (e.g. 182126)
* Score-8': number of users who scored 8. (e.g. 131625)
* Score-7': number of users who scored 7. (e.g. 62330)
* Score-6': number of users who scored 6. (e.g. 20688)
* Score-5': number of users who scored 5. (e.g. 8904)
* Score-4': number of users who scored 4. (e.g. 3184)
* Score-3': number of users who scored 3. (e.g. 1357)
* Score-2': number of users who scored 2. (e.g. 741)
* Score-1': number of users who scored 1. (e.g. 1580)

# **References**

Abdu Musowir U (2023). Language Translation with Python: A Beginner’s Guide using googletrans library. [online] Medium. Available at: https://medium.com/@musowir\_u/language-translation-with-python-a-beginners-guide-using-googletrans-library-6df86223a7b5 [Accessed 22 Nov. 2024].

Amos, D. (2022). Python GUI Programming With Tkinter – Real Python. [online] realpython.com. Available at: https://realpython.com/python-gui-tkinter/ [Accessed Nov. 2024].

Andryw Marques (2024). Pandas DataFrame: Show All Columns/Rows | Built In. [online] builtin.com. Available at: https://builtin.com/data-science/pandas-show-all-columns [Accessed 21 Nov. 2024].

Berukoff, T. (2023). Conquer Missing Data: Harnessing Pandas’ Techniques for Enhanced Analysis. [online] Medium. Available at: https://medium.com/@taylorberukoff/conquer-missing-data-harnessing-pandas-techniques-for-enhanced-analysis-5df017327ea5 [Accessed 22 Nov. 2024].

Cherry, K. (2023). How Does the Color Green Impact Mood and Behavior? [online] Verywell Mind. Available at: https://www.verywellmind.com/color-psychology-green-2795817 [Accessed Nov. 2024].

chiang, J. (2022). Overview of Item-Item Collaborative Filtering Recommendation System. [online] Geek Culture. Available at: https://medium.com/geekculture/overview-of-item-item-collaborative-filtering-recommendation-system-64ee15b24bb8 [Accessed 18 Nov. 2024].

Elsayed, M. (2024a). Mohamed Elsayed on LinkedIn: #dataanalysis #datascience #webscraping #machinelearning #plotly #dash… | 23 comments. [online] Linkedin.com. Available at: https://www.linkedin.com/posts/mohamed-elsayed-319a24204\_dataanalysis-datascience-webscraping-activity-7189033970657857537-K1Ay/ [Accessed 24 Nov. 2024].

Evelyn (2023). Collaborative Filtering in Recommender System: An Overview. [online] Medium. Available at: https://medium.com/@evelyn.eve.9512/collaborative-filtering-in-recommender-system-an-overview-38dfa8462b61 [Accessed Nov. 2024].

Gaurav, P. (2023). Step By Step Content-Based Recommendation System. [online] Medium. Available at: https://medium.com/@prateekgaurav/step-by-step-content-based-recommendation-system-823bbfd0541c [Accessed Nov. 2024].

GeeksforGeeks (2018). Python | Pandas Series.str.get\_dummies(). [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/python-pandas-series-str-get\_dummies/ [Accessed 24 Nov. 2024].

GeeksforGeeks (2024). How To Clear Out A Frame In The Tkinter? [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/how-to-clear-out-a-frame-in-the-tkinter/ [Accessed 19 Nov. 2024].

Itamar Turner-Trauring (2019). Reducing Pandas memory usage #1: lossless compression. [online] Python⇒Speed. Available at: https://pythonspeed.com/articles/pandas-load-less-data/ [Accessed Nov. 2024].

Mahmoud2227 (2024). GitHub - Mahmoud2227/IMDB-Dashboard. [online] GitHub. Available at: https://github.com/Mahmoud2227/IMDB-Dashboard [Accessed Nov. 2024].

Mohamed Elsayed (2024b). IMDB Data Analysis Dashboard. [online] mahmoud2227.pythonanywhere.com. Available at: https://mahmoud2227.pythonanywhere.com/ [Accessed Nov. 2024].

Plotly (2016). Layout.xaxis. [online] Plotly.com. Available at: https://plotly.com/python/reference/layout/xaxis/#layout-xaxis-showline [Accessed Nov. 2024].

Plotly (2024a). Dash in 20 Minutes Tutorial | Dash for Python Documentation | Plotly. [online] dash.plotly.com. Available at: https://dash.plotly.com/tutorial [Accessed Nov. 2024].

Plotly (2024b). Part 2. Layout | Dash for Python Documentation | Plotly. [online] dash.plotly.com. Available at: https://dash.plotly.com/layout [Accessed Nov. 2024].

Plotly (2024c). Plotly Dash App Examples. [online] plotly.com. Available at: https://plotly.com/examples/ [Accessed Nov. 2024].

Plotly (2024d). Setting the Font, Title, Legend Entries, and Axis Titles. [online] plotly.com. Available at: https://plotly.com/python/figure-labels/ [Accessed Nov. 2024].

Python Software Foundation (2024). tkinter — Python interface to Tcl/Tk. [online] Python documentation. Available at: https://docs.python.org/3.13/library/tkinter.html [Accessed Nov. 2024].

Saumyab271 (2021). Item-based Collaborative Filtering. [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2021/05/item-based-collaborative-filtering-build-your-own-recommender-system/ [Accessed 18 Nov. 2024].

scikit-learn developers (2024). TfidfVectorizer. [online] scikit-learn. Available at: https://scikit-learn.org/1.5/modules/generated/sklearn.feature\_extraction.text.TfidfVectorizer.html [Accessed Nov. 2024].

The Pandas Development Team (2024). pandas.Series.str.contains — pandas 1.4.2 documentation. [online] pandas.pydata.org. Available at: https://pandas.pydata.org/docs/reference/api/pandas.Series.str.contains.html [Accessed Nov. 2024].

Valdivieso, H. (2020). Anime Recommendation Database 2020. [online] Kaggle.com. Available at: https://www.kaggle.com/datasets/hernan4444/anime-recommendation-database-2020?select=anime.csv [Accessed 13 Nov. 2024].

W3Schools (2024). Pandas DataFrame iterrows() Method. [online] www.w3schools.com. Available at: https://www.w3schools.com/python/pandas/ref\_df\_iterrows.asp [Accessed 18 Nov. 2024].

Zach (2022). Pandas: How to Check if Column Contains String. [online] Statology. Available at: https://www.statology.org/pandas-check-if-column-contains-string/ [Accessed Nov. 2024].