Assessment Submission Cover Sheet

This Assessment Cover Sheet **must** be included on all Assessment submissions.

|  |  |
| --- | --- |
| Assignment Title | CA2 – Text Mining |
| Module | Data Mining |
| Student Name  (same as Student Card) | Ciaran Finnegan |
| Student Number | D21124026t |
| Programme |  |
| Part-Time/Full-Time |  |
| Year of Study  (First Year, Second Year, etc) |  |

Late Submissions: Assessment submitted after the deadline will have a late penalty applied.

**Academic Integrity for assessment in TU Dublin Programmes**

Each student is responsible for knowing and abiding by TU Dublin Academic Regulations and Policies. Any student in breach of these regulation/policies will be subject to action in accordance with the University’s procedures for breaches of assessment regulations. Please refer to the General Assessment Regulations at

<https://tudublin.libguides.com/c.php?g=674049&p=4794713>

<https://www.tudublinsu.ie/advice/exams/breachesofregulations/>

All students are expected to complete their courses/programmes in compliance with University regulations. No student shall engage in any activity that involves attempting to receive a grade by means other than honest effort, for example:

1. No student shall complete, in part or in total, any examination or assessment for another person.
2. No student shall knowingly allow any examination or assessment to be completed, in part or in total, for themselves by another person.
3. No student shall plagiarise or copy the work of another and submit it as their own work.
4. No student shall falsify any data. Falsification is the invention of data, its alteration, its copying from any other source, or otherwise obtaining it by unfair means, or inventing quotations and/or references.
5. No student shall use aids or devices excluded by the lecturer in undertaking course work or assessments/ examinations.
6. No student shall knowingly procure, provide, or accept any materials that contain questions or answers to any examination or assessment to be given at a subsequent time.
7. No student shall provide their assignments, in part or in total, to any other student in current or future classes of this module/ programme unless authorised to do so by the lecturer.
8. No student shall submit substantially the same material in more than one module/programme without prior authorization.
9. No student shall alter graded assignments or examinations and then resubmit them for regrading, unless specifically authorised to do so by the lecturer.
10. All programming code and documentation, unless correctly referenced, submitted for assessment or existing in the student’s computer accounts must be the students’ original work or material specifically authorized by the lecturer.
11. Collaborating with other students to develop, complete or correct course work is limited to activities explicitly authorized by the lecturer.
12. For all group assignments, each member of the group is responsible for the academic integrity of the entire submission. Consequently, all group members must satisfy themselves that all elements of their submission adhere to the academic integrity statement points above.

By submitting coursework, either physically or electronically, you are confirming that it is your own work (or, in the case of a group submission, that it is the result of joint work undertaken by members of the group that you represent) and that you have read and understand the University’s Regulations and Policies covering Academic Integrity (see General Assessment Regulations)*.*

Coursework may be submitted to an electronic detection system in order to help ascertain if any plagiarised material is present. If you have queries about what constitutes plagiarism, please speak to your lecturer.

|  |  |
| --- | --- |
| Student Signature |  |
| Date |  |

IMPORTANT:

* Complete the required number of tasks as defined in Assessment Handout
* The sections listed below are an example of the section headings for each task. You can use alternative headings
* Tasks 1-3: Sub-Sections 1-7 should be no longer than 8 pages (minimum 6 pages), including diagrams, images, screen captures, tables, etc. Careful selection of these is needed.
  + Code does not count to this total. Code should be added to the relevant section.
* Detailed discussion is expected. Marks are awarded based on depth of information given.
* Marks are awarded based on complexity of problem and depth of work.

# TASK 1 – *Text Mining – Comparison of Rotten Tomatoe Movie Reviews in Word Clouds*

1. **Definition of Problem**

Rotten Tomatoes is a very well know movie database website that allows critics and the public (defined as ‘Users’ in this assignment) to upload reviews and scores on new movie releases.

It has often been claimed that film critics are out of touch with the taste of regular movie goers and that this can be seen by the dichotomy between critic and user scores on Rotten Tomatoes, particularly for the larger commercial (‘blockbuster’) movie titles.

This assignment attempts to provide some data analytical rigour to this assertion of ‘aloofness’ by movie critics. Although one can just compare the scores given by critics and users to movies, this assignment attempts to identify noticeable differences in the patterns of the language used in the reviews themselves.

Review data from four of the most recent movies in the Marvel Cinematic Universe is pulled from the Rotten Tomatoes website, and Text Mining techniques are used to present opposing WordClouds built from the critic and user reviews.

Although it is only a sample of four movies, this Word Cloud analyse will provide a visual indication of the disparity, if any, between the reviews from critics and users.

1. **Data Exploration & Descriptive Analytics**

For simplicity, the assignment focuses on the Word Cloud analysis of the reviews for the Marvel movie ‘Eternals’ in particular. However, the Python code is built in a generic way to extract and analyse the review data from any given move on the Rotten Tomatoes website.

*Calling APIs*

For reasons explained in Section 6 of this section of the report, the data extraction from Rotten Tomatoes was carried out via the publicly available API.

Based on technical specifications and code snippets from a StackOverflow article**[1]**, two functions were written to separately extract the critics and the users movie reviews.

Text

Description automatically generatedFig 1 – Setting up ‘header’ variable for API Call

Text

Description automatically generatedFig 2 – Python Function to Extract Movie Reviews from Critics for ‘Eternals’

The function to extract the User reviews differs only in the API line.

Fig 3 – Code Snippet to Invoke APIS for User Reviews

I borrowed heavily from publicly available code to call the API call because of the challenges of web scrapping that were impeding my progress on the assignment.

Obviously, it would be better programming practice to have a single function to extract reviews and parse the API call using a flag in the function parameter. For several reasons, I kept the functions separate, despite the duplication in code:

* Simplicity. The primary focus of this assignment is intended to be on setting optimal parameters for the Word Cloud display and I wanted to move quickly to that stage of development.
* The User reviews are considerably larger in volume than the review data from the Critics and take a significant amount of time to extract, usually in the order of several minutes. I experimented with different ‘Sleep()’ values for both sets of review for ‘Eternals’.

*Working with Dataframes*

The output from the APIs populates dataframes with a collection of data attributes for a given movie.

The dataframes returned by the APIs differ in structure between Critics and Users.

Graphical user interface, text, application, email

Description automatically generatedFig 4 – Dataframe for Critics Reviews

Better Python coding practice would have been to set up a dictionary of the movie URLs and run the process through a loop to generate WordClouds. For simplicity in this assignment, I just created one Jupyter Notebook per movie.

Table

Description automatically generated with medium confidenceFig 5 – Dataframe for User Reviews

Running a simple line of code pulls out and concatenates all the review text from each dataframe.

The ‘quote’ column contains the text from Critic reviews for ‘Eternals’.



Fig 6 – Extract and Concatenate Critic Reviews into Text Variable

The ‘review’ column contains the text from User reviews for ‘Eternals’.



Fig 7 – Extract and Concatenate User Reviews into Text Variable

These text variables then form the starting point of the data preparation phase in the following section.

1. **Data Preparation**

**NLTK**, or Natural Language Toolkit, is a Python package that we will use to  carry out **text pre-processing**tasks before creating the WordClouds.

*Tokenize the Words*

The first step in the data preparation process is to Tokenize the text. This begins the process of turning unstructured data into structured data, which will be easier to analyse and present in WordClouds. In our Python code we begin by breaking the critic and user reviews into individual words.

Graphical user interface, text, application, email

Description automatically generated

Fig 8 – Critic Reviews Tokenized

A picture containing table

Description automatically generated

Fig 9 – Partial View of Tokens for Critic Reviews

Graphical user interface, text

Description automatically generated

Fig 10 – User Reviews Tokenized

Note, the significantly greater volume of User reviews, which took many minutes to extract.

*Remove Numbers and Punctuation: Filter Tokens*

The text of the reviews only needs to contain words. Numbers and punction will be of no value in the analysis we present in the WordClouds.

The following code will filter the tokens and eliminate the text that is not required. It is also good practice to reduce all characters to lower case.

Graphical user interface, text, application, chat or text message

Description automatically generated

Fig 11 – Critic Reviews Filtered for Punctuation and Numbers

Table

Description automatically generated with medium confidence

Fig 12 – Critic Reviews – Filtered (sample)

Graphical user interface, text, application, chat or text message

Description automatically generated

Fig 13 – User Reviews Filtered for Punctuation and Numbers

Again, we significant disparity in word volume.

*Setup Stop Words*

**Stop words** are words that should be to ignored and will be filtered out in this stage of text pre-processing.

Very common words like 'in', 'is', and 'an' are often used as stop words as they offer little meaning to a text in and of themselves.

The Python NLTK library contains several pre-defined English stop words, which shall be removed first.

Graphical user interface, text

Description automatically generated

Fig 14 – Set Up Standard English ‘Stop Words’

The quality of the analysis presented by the WordClouds will be improved by adding additional stop words to the list. In this assignment, as the analysis focused on Marvel movies, both sets of reviewers would use words like ‘*mcu’*, *’film’*, *’universe’*, *’comic’* and so on that did not provide useful context or differentiation. The title of the film was also the most common phrase in both sets of reviews so that was included in the stop word list.

The final list of custom stop words was arrived after an iteration process that generated WordClouds and identified words and phrase that added little value to the data visualisation.

Scatter chart

Description automatically generated

Fig 15 – Add Additional Assignment Specific ‘Stop Words’

These steps reduce the number of tokens in each set of reviews.

Graphical user interface, text, application

Description automatically generated

Fig 16 – Filter Stop Words from Critic Review Text: 4075 tokens remain

Graphical user interface, text, application

Description automatically generated

Fig 17 – Filter Stop Words from User Review Text: 326055 tokens remain

1. **Configuration of WordCloud Parameters**

At this point in the assignment the next step is to create a WordCloud for each finalised data set of tokenised words for reviews. The WordCloud Python library will create the WordClouds, which will then be displayed as side-by-side subplots by the matplotlib library.

*Shaping the WordClouds*

To make the WordClouds more visually appealing, the shape of the display is being converted from a square into a cloud outline (the image could also be interpreted as a ‘think bubble’).

The Critics reviews ‘cloud’ will be presented in the upper part of the subplot display in black. The User Reviews will render in red using an inverted cloud image as the outline.

To create the ‘mask’ for the WordCloud function parameter, two *png* images were created and added to the assignment file directory.

A picture containing bowed instrument, spectacles

Description automatically generatedA pair of sunglasses

Description automatically generated with medium confidence

Text

Description automatically generated with medium confidence

Fig 18 – Outline Images to Shape WordClouds

The application of these masks is explained in the following section on WordCloud function parameters.

*Configuring the WordClouds*

The initial objective of the WordClouds in this assignment was to give visual prominence to words most frequently used by Critics and Users and compare the displays.

Single words did provide some interesting comparisons but additional research on WordClouds generated from the text of speeches in recent American political campaigns**[2]** indicated that more value might be extracted if short phrases were included. Therefore, the parameters were set on the WordCloud to look at bigrams (tokens more frequently seen in sequence).

The configuration for the Critic Review WordCloud function is given in the following diagram:

Graphical user interface, text

Description automatically generated

Text, letter

Description automatically generated

Fig 19 – Critic Reviews: WordCloud Function Parameters

The colour of the Critics review WordCloud is determined by the black colour of the mask used.

A picture containing diagram

Description automatically generated

Fig 20 – Critic Reviews: Setting the Colour

The parameters for the User reviews WordCloud function are identical, with the exception of another line to alter the colour to ***red***.



Fig 21 – User Reviews: Setting the Colour

The key WordCloud function parameters, and their purpose in this assignment, can be briefly described as;

* ***collocations***=True : Allows two-word phrases to be displayed. The result of this is evident in the final WordCloud display.
* **min\_word\_length**=4 : Avoids smaller less meaningful words/phrases
* **collocation\_threshold**=3 : This sets a threshold above which a bigram must score to be considered for the WordCloud

The ‘*collocation\_threshold’* was a value and concept that was somewhat challenging to comprehend but, through trial and error, it did help generate interesting visual comparisons in the review WordClouds.

1. **WordCloud Displays**

Some simple Python code rendered the WordClouds for Critic and User reviews of the Marvel movie ‘Eternals’.

Graphical user interface

Description automatically generated with medium confidence

Fig 22 – Render and Format Position of Review WordClouds

The resulting sub-plots provide interesting insight into the differences between the reviews.

A close-up of a sign

Description automatically generated with low confidence

Text

Description automatically generated

1. **Comparison with other Research & Reflections**

*1 – Presenting Opposing WordClouds*

This Text Mining assignment had originally intended to look at the difference in language tone across news websites covering the same, or similar, political events. Although the approach and subject matter of the assignment changed significantly, there were some research papers relating to Text Mining of news outlets that continued to provide guidance on the use and presentation of WordClouds.

Of note was an article written by in 2012 Hensinger, Flaounas and Cristianini at Bristol University entitled *The Appeal of Politics on Online Readers***[3]**. The authors employed an opposing set of WordClouds to show what words most and least appealed to readers in terms of their likelihood to read an article in Forbes magazine.

This technique seemed to fit well with the objective of this assignment, and I followed their approach to generate a single contrasting ‘at-a-glance’ view, from a Rotten Tomatoes movie page, of both the critic and user review WordClouds.

*2 – Bigrams and Working with Python WordCloud Parameters*

Although the initial WordClouds generated in this assignment were providing a reasonable sense of review content, the focus on displaying single words was removing some valuable context. Looking at related research on sentiment analysis from Rotten Tomatoes there was a reference in the paper from Sorostinean, M., Sana, K., Mohamed, M. and Targhi, A., 2017. *Sentiment Analysis on Movie Reviews***[4]** that recommended the use of Bigrams. (A bigram is a sequence of two adjacent elements from a string of tokens.)

Researching the use of Python WordClouds, I found a further article on the *towardsdatascience.com* website entitled *Generate Meaningful Words in Python***[2]**. This provided some practical examples on settings for the parameters in a Python WordCloud so that I could extract meaningful short phrases from the reviews, along with single words.

The recommended setting of eliminating words with fewer than four characters and then setting the collocation threshold to ‘3’ produced more meaningful WordClouds. This is described in more detail in Section 4 of this section of the assignment.

I am conscious that I am greatly simplifying the description of the research output to which I referred, and that there was also an element of trial and error in the settings, but I felt the

Rotten Tomatoes WordCloud output was very satisfactory.

*3 – Calling APIs vs. Web Scrapping*

My initial Python code used the *BeautifulSoup* library to scrape the data directly from the Rotten Tomatoes web page for the given Marvel movie.

This has an immediate limitation in that it only returned the text for the reviews visible on the first web page. There were several online resources that provided guidance on URL manipulation as a possible solution. However, the Rotten Tomatoes website is constantly evolving in terms of its structure and the *BeautifulSoup* approach was not proving very robust in the face of these changes.

Looking over online research papers in movie reviews and text mining, I found that others had begun their projects**[5]** by using the publicly available APIs that websites such as Rotten Tomatoes provide for data retrieval. Following this trail of breadcrumbs led to online technical specifications and sample code that greatly simplified the extraction of Rotten Tomatoes review data.

*Reflections*

The inclusion of short two-word phrases helps show, through our WordClouds, that critics tend to focus on directors and more technical aspects of the movie, while users are more direct with their emotional enthusiasm.

Our ‘Eternals’ WordCloud for Users was able to capture phrases like ‘*really good’* and ‘*really enjoyed’*. This type of phrase is conspicuous by its absence in the critic reviews, although I did enjoy reading the phrases ‘*colossal bore’* and ‘*glacial pace’* buried in the Critics WordCloud.

For the film ‘Eternals’ the Users were clearly more positive than the Critics, although this pattern is less obvious for the other movies, which can be seen in the attached WordCloud movie images below.

**

For example, Users seem to make more of an issue around the plot of ‘Endgame’ than was the case with the Critics.

1. **References**

[1] Adriaansen, R., 2021. *Scraping all reviews of a movie from Rotten Tomato using soup*. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/69963743/scraping-all-reviews-of-a-movie-from-rotten-tomato-using-soup> [Accessed 20 December 2021].

[2] Dickenson, B., 2020. *Generate Meaningful Word Clouds in Python*. [online] towardsdatascience.com. Available at: <https://towardsdatascience.com/generate-meaningful-word-clouds-in-python-5b85f5668eeb> [Accessed 20 December 2021].

[3] Hensinger, E., Flaounas, I. and Cristianini, N., 2012. *The Appeal of Politics on Online Readers*. [online] Blogs.oii.ox.ac.uk. Available at: <http://blogs.oii.ox.ac.uk/ipp-conference/sites/ipp/files/documents/HensingerFlaounasCristianini\_Oxford2012.pdf> [Accessed 20 December 2021].

[4] Sorostinean, M., Sana, K., Mohamed, M. and Targhi, A., 2017. Sentiment Analysis on Movie Reviews. *Journal Agroparistech.*, [online] Available at: <http://www.agroparistech.fr/ufr-info/membres/cornuejols/Teaching/Master-AIC/PROJETS-M2-AIC/PROJETS-2016-2017/main(Amal%20Targhi-%20Mihaela%20SOROSTINEAN-%20Katia%20Sana-Mohamed%20Mohamed).pdf> [Accessed 20 December 2021].

[5] Schaible, J., Carevic, Z., Hopt, O. and Zapilko, B., 2015. *Utilizing the Open Movie Database API for Predicting the Review Class of Movies*. [online] Citeseerx.ist.psu.edu. Available at: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1071.6147&rep=rep1&type=pdf> [Accessed 20 December 2021].

# TASK 2 - *<insert select Task Name here e.g. Association Rules Problem>*

1. **Definition of Problem**

Clearly state the problem definition, what type of data mining task is it, where was the data set sourced from, etc.

1. **Data Exploration & Descriptive Analytics**

Include any data insights discovered

1. **Data Preparation**

Include details of any data cleaning, transformations, data enrichment, feature engineering, feature reduction, etc

1. **Details of Algorithms & Configurations**
2. **Model Performance Metrics & Evaluation of Results**
3. **Comparison with other Research**

Compare your results to at least three other researchers (maximum of five) who used the same data set. What lessons did you learning from doing this? How can your work be improved? Did you include any improvements in your work and what impact did it have?

1. **References**

Use the IEEE Referencing style. See this guide for details. <https://libraryguides.vu.edu.au/ieeereferencing/gettingstarted>

# TASK 3 - *<insert select Task Name here e.g. Time Series Analysis Problem>*

1. **Definition of Problem**

Clearly state the problem definition, what type of data mining task is it, where was the data set sourced from, etc.

1. **Data Exploration & Descriptive Analytics**

Include any data insights discovered

1. **Data Preparation**

Include details of any data cleaning, transformations, data enrichment, feature engineering, feature reduction, etc

1. **Details of Algorithms & Configurations**
2. **Model Performance Metrics & Evaluation of Results**
3. **Comparison with other Research**

Compare your results to at least three other researchers (maximum of five) who used the same data set. What lessons did you learning from doing this? How can your work be improved? Did you include any improvements in your work and what impact did it have?

1. **References**

Use the IEEE Referencing style. See this guide for details. <https://libraryguides.vu.edu.au/ieeereferencing/gettingstarted>

# TASK 4 - *<insert select Task Name here e.g. Data Ethical Issues >*

## Task 4-1 : <Title of Case Study)

1. **Overview of problem**
2. **Ethical and Legal Challenges**
3. **Challenges for Data Scientist**
4. **Reflections**
5. **References**

Use one of the commonly used References and Citation formats.

## Task 4-1 : <Title of Case Study)

1. **Overview of problem**
2. **Ethical and Legal Challenges**
3. **Challenges for Data Scientist**
4. **Reflections**
5. **References**

Use one of the commonly used References and Citation formats.