Web crawling, data analysis, and data visualisation skills are applied to scrape the required website, store information in suitable formats, analyse different topics, and present relevant results. The report is organised following the sequence from question 1 to 5.

The overall logic of question 1 is to crawl all the detailed webpages from the publication page by topic and then drop the duplicated ones. First, the link of the publication page is retrieved by BeautifulSoup and ‘find’ function. Second, the url of the current topic—‘Character Animation’ is crawled from the current page, followed by the links of all the other topics retrieved from the ‘a’ classes under ‘TextOption’ class. Then, the duplicated webpage links are stored in a list using a nested iteration over the different topics and the specific classes where links are stored, the results of which are concatenated to the home link to make a complete url. Finally, unique links are first converted to a set to remove duplicates, and then stored in a sorted list corresponding to the original index so that they can be used as a reference in question 2.

In question 2, titles, publication webpage links, topics (up to 3), authors (up to 7), abstracts, journals, impact factors, the number of citations, and links for paper, video, DOI, and Youtube are crawled. The final output is written to a csv file. While it might seem straightforward to directly make use of the urls in question 1, the name of the topics that a publication belongs to only appears on the publication page rather than the detailed webpage. Therefore, pairs of duplicated webpage links and topics are stored, converted, and concatenated to a data frame with 2 columns instead (e.g., webpage, topics). In the duplicated information, only the second and the third topics are useful and are formed as two new columns. This process is realised by finding the unique indices, store the duplicated lines in a new data frame while dropping them in the original data frame, and outer merge the dropped line to the original data frame with unique lines on the key of webpage links. As there is only one publication with 3 topics, its third topic is written manually using ‘df.loc’ with a new column with nan values created. The output is checked against question 1 for consistency. While most other information can be found in the detailed webpage, some specific elements may not present, resulting in errors. Therefore, while they are all crawled in a ‘for loop’ over webpage links, ‘try’ and ‘except’ are applied to certain information to balance between robustness and convenience.

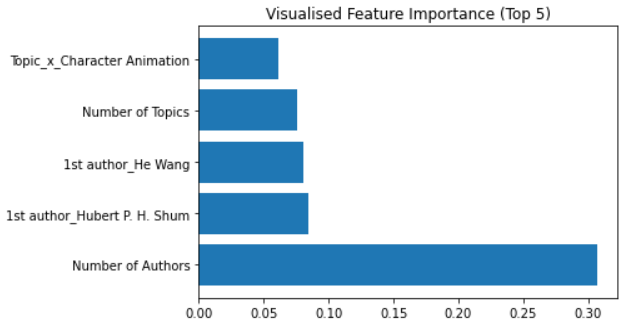
In question 3, top 100 popular words of one unit are selected and presented using TF-IDF algorithm. According to Cambridge Dictionary (n. d.), ‘word’ is defined as ‘a single unit of language that has meaning and can be spoken or written’. This report adopts this definition and defines a word to be of one sole unit, which keeps phrases out of the scope of this question. In order to extract words that are as meaningful as possible, the solution applies the TF-IDF algorithm, where the more frequently a word appears in a corpus, the higher a penalty is applied to the corresponding word. The score is calculated by multiplying term frequency, the number of times a word appears, by inversed document frequency, the inversed and log normalised number of documents containing the term in the corpus. In this way, the extract words should not only have meanings, but also correspond to the topics to which their publications belong. The function is realised using ‘nltk’ package and self-defined TF-IDF score functions. After storing all the abstracts and titles in one list, the texts are tokenised using ‘word\_tokenize’, with punctuation as well as stop words removed. Then, term frequency and inversed document frequency are calculated by their definitions and are applied to each word by multiplication. Finally, a descending sorted list of lists storing TF-IDF scores and terms is created and top 100 popular words are presented according to their scores.

The solution to question 4 regards exploratory data analysis on co-authorship. Specifically, top 10 authors of all topics, top 5 co-authors with the top 3 co-authors with Hubert, and top 10 co-authors with Hubert by top 3 topics are presented. The solution makes use of the data frame in question 2. Since the website is about Dr Hubert, the top 10 authors of all topics are in effect top 9 co-authors. For the same reason, top co-authors with his top co-authors excludes he himself as well as the author—the research object of an individual graph. While the overall author frequency is counted using ‘Counter’ from ‘collections’ by storing all authors in a list and converting the result back to a list, other calculations are a bit more complicated. For them, a new data frame is created with 2 new columns storing all topics and authors of a publication. Then, information corresponding to the current research object is selected using ‘df.loc’ and ‘df[“column”].str.contains(“research object”)’. Accordingly, counts are calculated within the corresponding data frame using ‘Counter()’. Specifically, counted dictionary values are mapped to lists and then stored in a list so that the index corresponds to the crawled webpage index. Finally, bar charts are plotted with x labels arranged vertically by ‘plt.xsticks(rotation=270)’ for clarity. The results are presented in table 1 below.

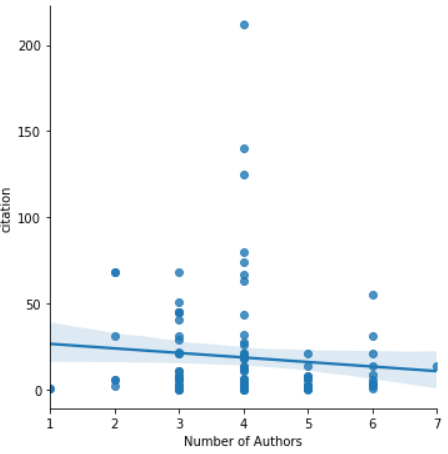
|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
|  |  |  |

*Table 1*

Overall, the solution to question 5 can be divided into two parts. The first part regards a random forest for feature selection utilising the feature importance and its preparation. I The second part concerns scatter plot and line of best fit corresponding to the features selected in the previous step. Random forest is based on classification and regression trees, the latter of which make splits that minimise deviance. Such methods make no prior assumptions on the dataset and are thus selected. Random forest improves CART methods by selecting a subset of features in each tree as well as bagging (James et al, 2021). Feature importance is calculated via ‘.feature\_importances\_’ function from scikit learn, which derives from the mean accumulated decreased impurity within a single tree (Scikit Learn, n.d.). In practice, 4 predictor variables are chosen for random forest feature selection (e.g., number of topics, number of authors, topic\_x, 1st author). Although the impact factor intuitively serves as a promising indicator for citation, it is dropped for too many na values. The five variables are stored in a data frame, filled with 0s for na values, and coded to dummy variables by ‘pd.get\_dummies’. Then, a classification random forest is built using ‘RandomForestClassifier’, after splitting training and test sets by ‘train\_test\_split’. Test accuracy is 77.27% calculated by ‘metrics.accuracy\_score’, which indicates a reasonably good fit. Top 5 feature importance is subsequently pulled and plotted as presented in figure 1. It can be seen that only the importance of the number of authors exceeds 0.1, which makes it the only feature selected. Finally, a scatter plot of citation against number of authors is plotted along with the line of best fit as shown in figure 2. It is interesting to note that while some literature suggests that papers by more authors are likely to have higher citations as authors tend to cite the works from whom they co-authored with (Yan et al, 2011), there appears a slightly negative correlation between the two variables here. This may be due to the limited datapoints in the training set, which can be indicated by the relatively sparse scatters in 1 and 2, and the much denser points in 5 and 6.



*Figure 1. Top 5 Feature Importance*

**

*Figure 2. Line of Best Fit*

In conclusion, question 1 stores unique webpage link in a list which question 2 utilises for consistency check. Question 2 stores text-based information in a data frame with individual topic and author in separate columns, and is saved to a csv file. Question 3 utilises TF-IDF. Question 4 presents most popular co-authors by author and topic. Question 5 selects features by random forest feature importance, and plot correlation between the two variables, finding a negative relationship.

References

Cambridge Dictionary (n. d.) *word*. Available at: https://dictionary.cambridge.org/dictionary/english/word (Accessed: 15th Feb 2022)

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2021) *An Introduction to Statistical Learning with Applications in R*. 2nd edn. Springer

Scikit Learn (n. d.) *Feature importances with a forest of trees*. Available at: https://scikit-learn.org/stable/auto\_examples/ensemble/plot\_forest\_importances.html (Accessed: 15th Feb 2022)

Yan, R., Tang, J., Liu., X., Shan., D., and Li., X. (2011) ‘Citation Count Prediction: Learning to Estimate Future Citations for Literature’, *CIKM '11: Proceedings of the 20th ACM international conference on Information and knowledge management.* pp 1247-1252

doi.org/10.1145/2063576.2063757