COMP42315 Assignment – Z019003 Programming for Data Science

# Question 1:

I have designed a web-scraping algorithm using Python's BeautifulSoup and pandas libraries to extract publication titles along with their citation numbers in descending order. The algorithm operates through the following steps:

Initialization, Web Scraping, Publication URLs Extraction, Data Extraction, Descending Sort, Visualization

In the Initialization phase, we set the groundwork by preparing an empty list meant for storing publication page URLs.

Moving to Web Scraping, the core task was extracting topic names and their respective URLs. This was achieved by parsing the HTML content from the main URL with BeautifulSoup.

With the topic URLs in hand, the next focus was on Publication URLs Extraction. For each topic URL, additional GET requests were made, revealing the URLs of individual publications.

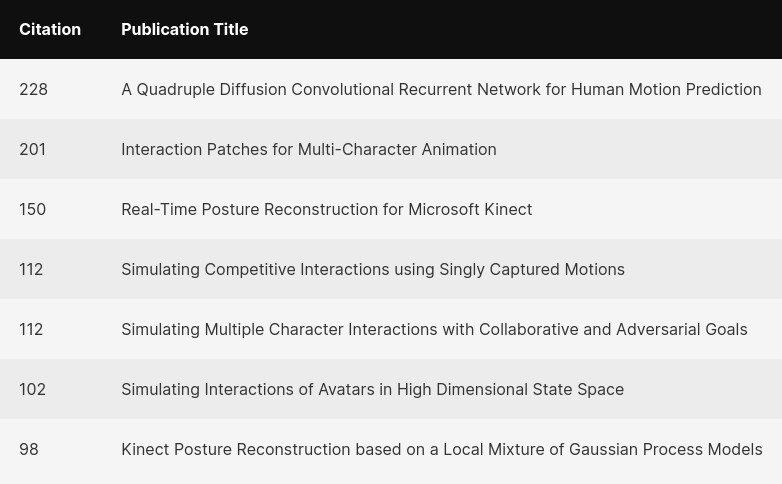
Data Extraction marked a deeper dive. Each publication URL received a GET request, and from the resulting HTML content, title and citation were extracted and stored in a DataFrame.

Post-collection, the data needed structure. In the Descending Sort step, the DataFrame was rearranged based on citation numbers, placing the most cited publications at the forefront.

In our recent data analysis, we've crafted a table to elegantly present our publication insights. This table, however, is distinct; it highlights the top seven pivotal publications.

The metric we've leaned on to gauge their prominence is the frequency of citations received from scholar peers.

## Table 1: Publication Titles by number of citations



Within the research domain, citations serve as endorsements from scholarly peers. A publication that garners numerous citations indicates its intrinsic value and relevance. Thus, these citation figures aren't merely statistical data.

Word Count: 250

**Question 2:**

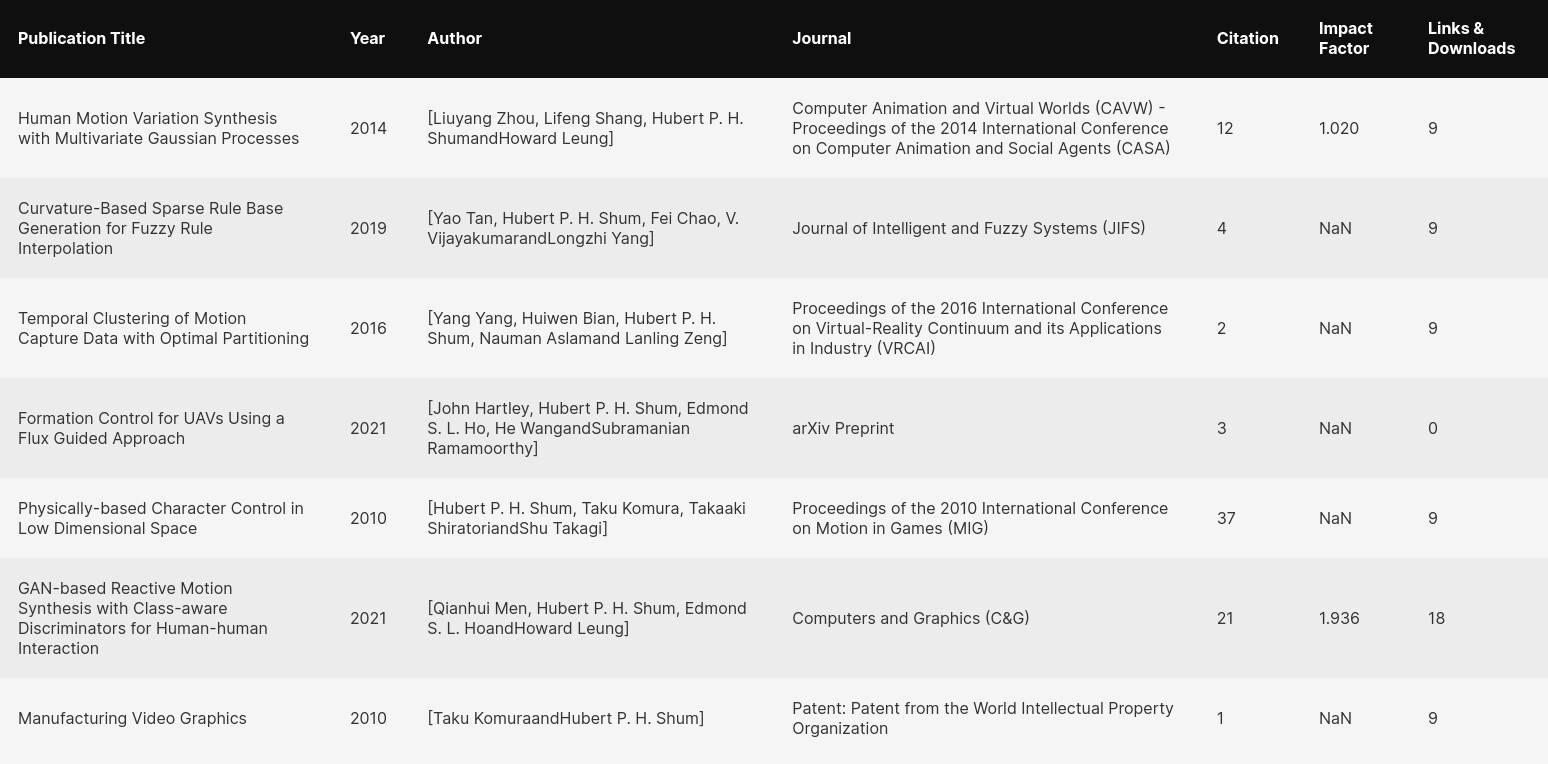
With the scrape\_and\_count\_features function, we're able to capture essential publication details. These encompass the title of the publication, the year it was released, its authors, the journals it appeared in, the citation count, its impact factor, and the tally of its link downloads.

Once extracted, these details are thoughtfully organized into lists. To make this process operational, the function requires an initial input: a list of URLs.

Furnished with this, the function meticulously navigates each URL, pinpointing the exact elements that store our sought-after information. Each piece of data is then cataloged in the corresponding lists.

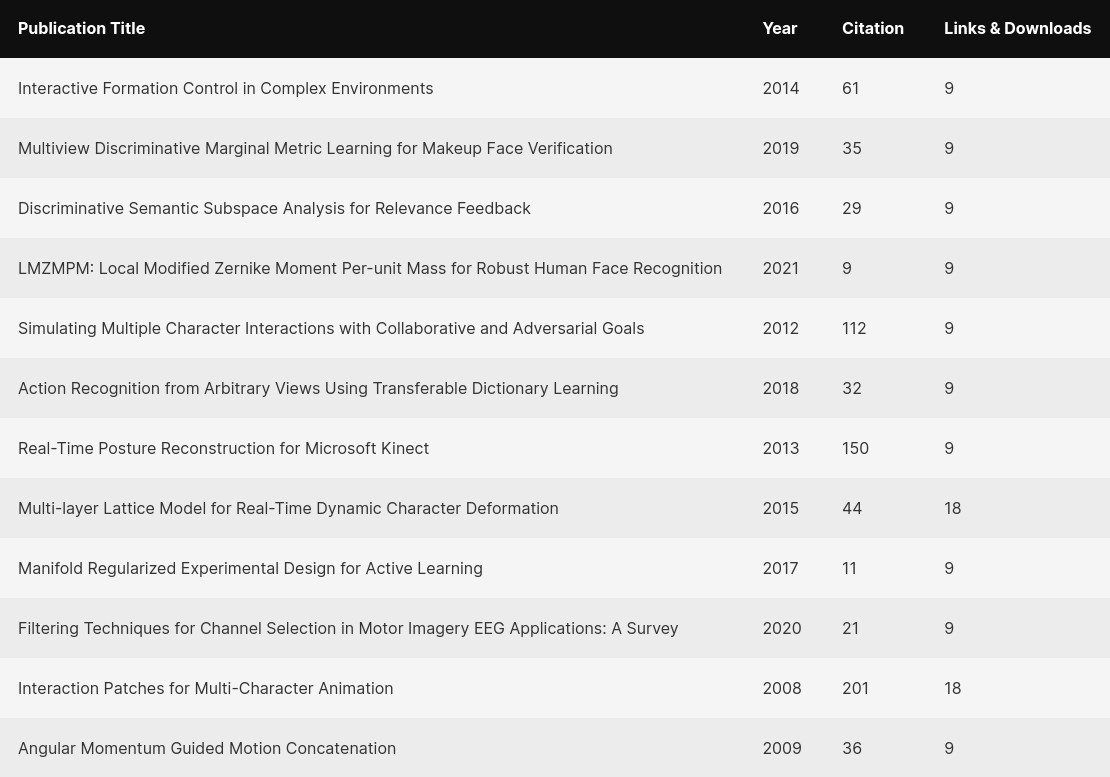
After all the URLs have been combed through, the end product is a cohesive DataFrame brimming with the gathered publication details.

## Table 2: Publication Titles by year, author, citation, impact factor…

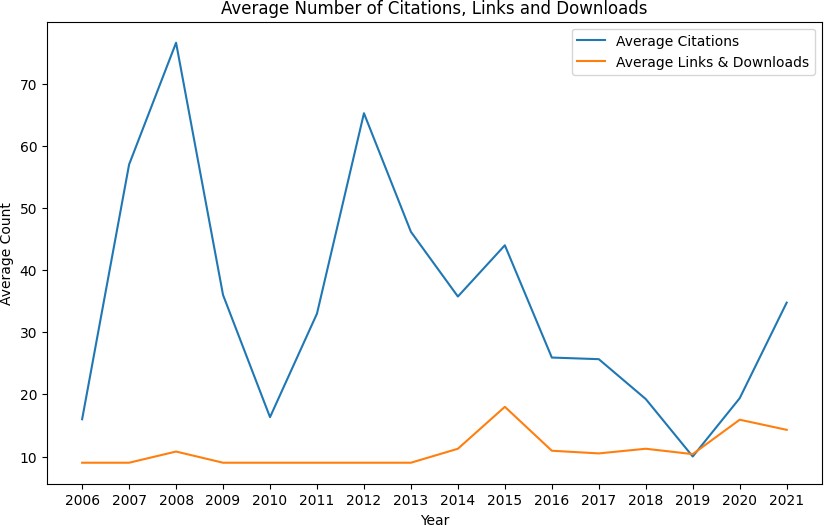


By calling upon the scrape\_and\_count\_features function, publications are ranked in descending order. Specifically, for each calendar year, we spotlight the publication with the highest impact factor. From these publications, we detail out the title, its publication year, the amassed citations, and the count of links and downloads.

## Table 3: Publication Titles by year, citations, links&downloads



To provide a visual representation of the average citation, link and download trends, the code generates a plot using Matplotlib.



## Figure 1: Moving Average citations and Moving Average Links&Downloads

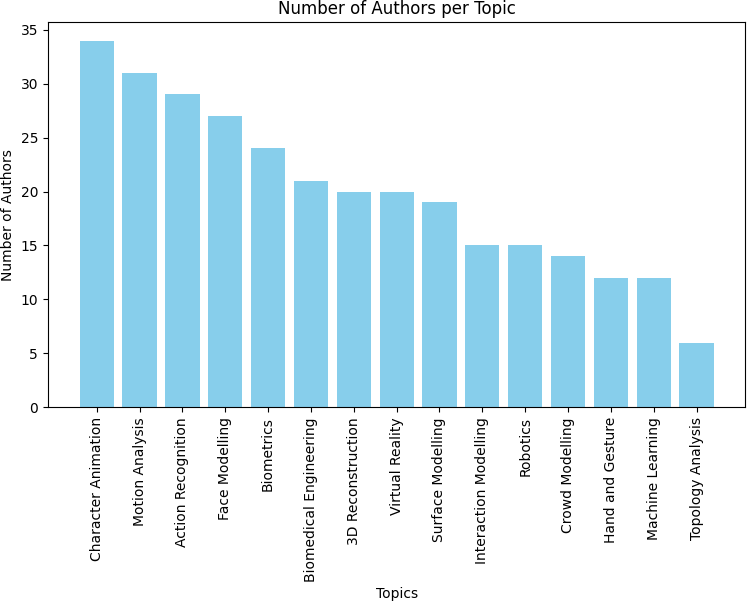
In a nutshell, the average count of citations follows a different dynamic than links and downloads. On the one hand, there has been only one surge of the number of downloads (and click counts on hyperlinks) in 2015. On the other hand, there is a sinusoidal pattern for the average count of citations with surges in 2006, 2010 and 2019.

Word Count: 250

**Question 3:**

**a.**

Initially, we crafted an empty dictionary titled topic\_author\_counts. This dictionary's primary role was to house topics and their respective author numbers. Each URL was then analyzed, and its publication divs were scrutinized to understand author collaboration patterns. This process allowed for regular updates to the author count for each topic within the topic\_author\_counts dictionary. By harnessing the power of the max() and min() functions, we pinpointed topics with the most abundant and scantiest author counts, respectively. For a lucid representation of our findings, the topic\_author\_counts dictionary was arranged in descending order. Topics and their author counts were subsequently extracted separately.



## Figure 2: Number of authors by topic

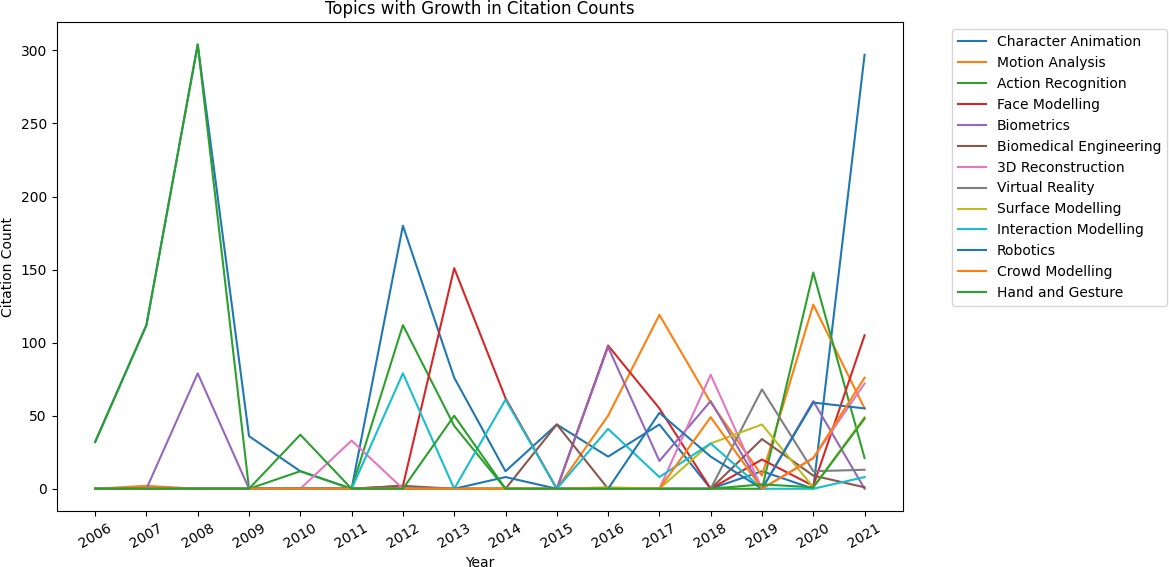
A bar chart was then fashioned using matplotlib, showcasing the number of authors attributed to each topic.

Notably, "character Animation" boasts the most authors, while "Topology Analysis" features the least.

"Character Animation" stands out with the highest number of authors, while "Topology Analysis" has the fewest.

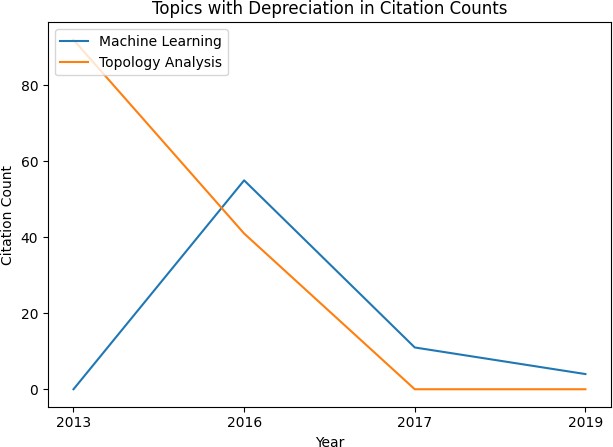
## b.

To discern the ebb and flow of citations over time, we turned to the topics\_citations\_years\_count dictionary. Through this, we identified topics experiencing growth or decline in their citation counts. We then visualized these trends leveraging Matplotlib.



## Figure 3: Temporal evolution of number of citations by topic

Our journey to trace citation patterns led us to utilize the topics\_citations\_years\_count dictionary. It served as our tool to recognize topics that saw a rise or dip in citations. With matplotlib at our disposal, these patterns were vividly visualized, encapsulating the citation flux across years.



## Figure 4: Comparative evolution of citation count for ML and topology analytics

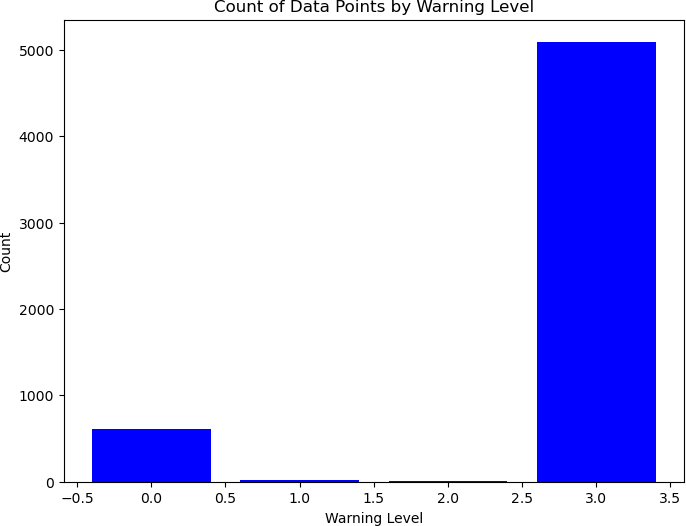
Finally, a comparative analysis between the least cited topics, i.e., Machine Learning and Topology, shows that both had different trajectories before 2016 but this year has marked the start of a significant drop for Topology citations in 2017 with a downward trend continuing until 2019.

Word Count: 273

**Question 4:**

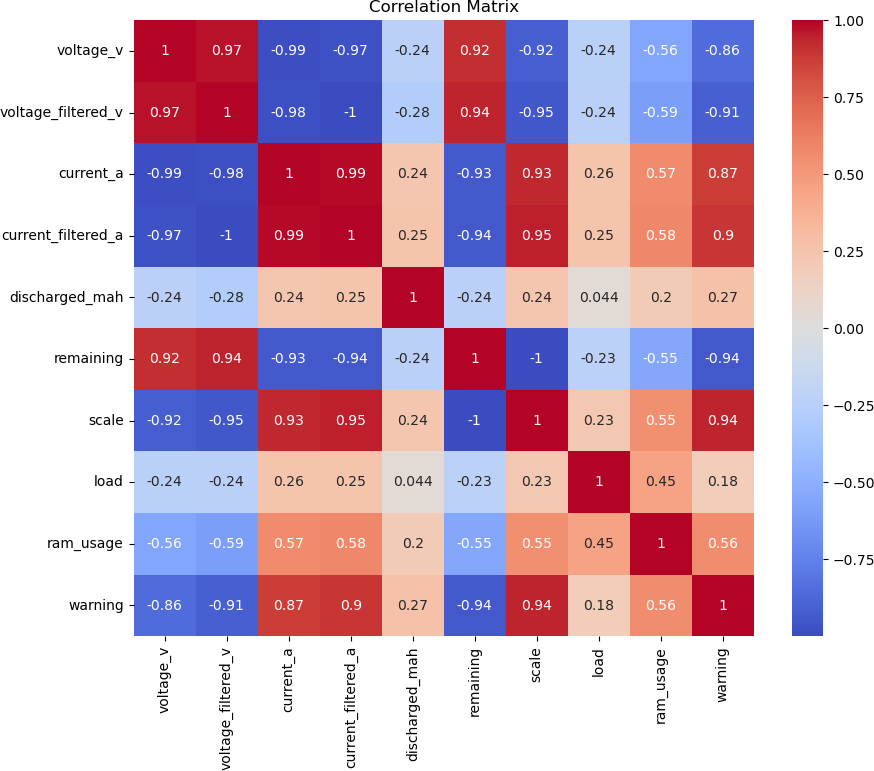
To understand the relationship between attributes and the warning level, we can use the correlation matrix, scatter plots and boxplots.

First, we calculate the sum of each warning level:

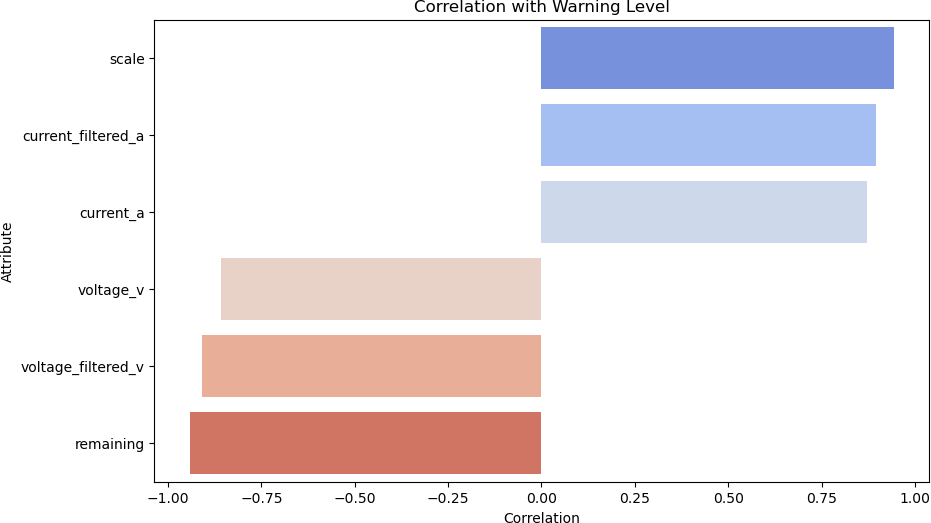


## Figure 5: Warning Level count

Then we can Plot a heatmap of the correlation matrix:



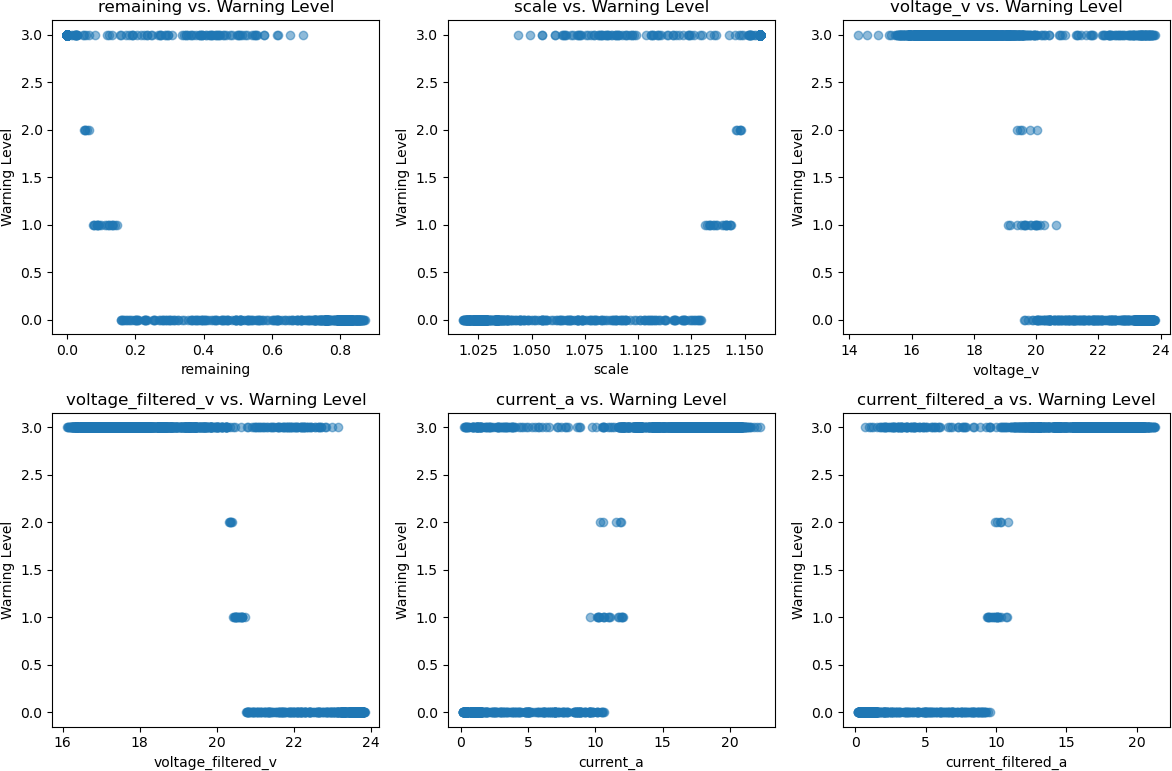
## Figure 6: Attributes’ Correlation Matrix



**Figure 7: Correlation with warning level for each attribute**

We have now highlighted and visualize attributes that have the highest probabilistic relationship with warning.

We can create scatter plots for each attribute:



## Figure 8: Attributes vs warning Scatterplot

The attributes 'voltage\_v', 'voltage\_filtered\_v', and 'remaining' exhibit negative correlations with the 'warning' level. This suggests that as these values rise, the warning level likely drops. Essentially, an uptick in voltage or the 'remaining' attribute correlates with a decline in the warning level.

Conversely, 'scale', 'current\_a', and 'current\_filtered\_a' demonstrate positive correlations with the 'warning' level. This insinuates that as these values go up, the warning level also escalates. Put simply, an increase in these current values prompts a rise in the warning level.

Lastly, 'discharged\_mah', 'load', and 'ram\_usage' showcase only faint correlations with the 'warning' level. It's probable that these variables don't share a robust linear association with the 'warning' level.

As a conclusion, the attributes that show a statistically significant and either positive or negative relationship with the 'warning' level are 'remaining', 'scale', 'voltage\_v', 'voltage\_filtered\_v', 'current\_a', and 'current\_filtered\_a'.

Word Count: 200

Total Word Count: 1000