Final Project Report

Data Mining CSC 84040

Fall 2019

Joanne Ramadani

M.S. in Data Analysis and Visualization

**Executive Summary**

This project focuses on the metadata of a fanfiction archive site (Fanfiction.net), finding patterns that arose given the relationship between a work’s word count and how much engagement it gets (measured in amount of reviews, “favs”, and follows). The primary purpose of this project is to see if there is a correlation between a work’s word and chapter count and level of engagement, and if so, what that correlation looks like in order to maximize engagement potential for those looking to learn how to write. In order to analyze this, I used clustering to understand patterns in word and chapter count as they relate to engagement levels, and found that in general, works that range from 60,000 to 80,000 words or 10-20 chapters tend to receive the most engagement.

**Background**

Fanfiction (colloquially referred to as “fic”) is a fan-written story about a piece of media—the story can be anywhere from 50 words to over 8,000,000 words (as the data ended up showing), usually taking the characters of the world and writing scenarios that are not present, or would otherwise be impossible, in the original work. This transformative work is undertaken by a variety of people, but particularly in the case of English Language Learners and young people learning to write, it can be a gateway to learning how to write effectively.

**Tools Used**

Throughout this project, I tested a few different tools (Azure’s Jupyter notebooks, Google Colabs, even the shell on my computer) before settling on Visual Studio Code, an IDE that combined my need to work locally due to the size of my data and how much I had to change throughout the process. I ended up primarily using the libraries NumPy, Pandas, MatPlotLib, and Scikit-Learn, particularly Scikit-Learn’s Mini Batch K-Means model.

**Process: Knowledge Discovery in Databases (KDD)**

1. **Pre-planning**

When creating this project, I had two main goals in mind: 1) figure out what kind of story had the most chance of receiving a large amount of engagement and 2) create a project that I knew would appeal to a large audience. Fanfiction isn’t necessarily a mainstream activity, but it’s popular enough that my dataset included approximately 5 million records, and a different archive has over 5.4 million works at the time of writing. On doing more research, I discovered that there was an educational aspect to it, but that educational aspect is derived through the act of receiving engagement from readers. If you are simply posting your creative work without a way for others to interact, it becomes like giving a speech to an empty room. Thus, what my project measures—engagement—directly correlates to the utility of fanfiction in an educational context.

1. **Dataset**

The dataset was made available on Kaggle by the user Metrovirus, and [the link can be found here](https://www.kaggle.com/metrovirus/fanfictionnet). It is a web scrape of all public fanfiction records available on the archive Fanfiction.net from its inception (unless the fiction was deleted at time of scrape) up until October 2018. Originally, the dataset contained 17 features across several thousand individual .csv files representative of a piece of media with fanfiction. Those .csv files were individually categorized by type of media (tv, movie, play, video game, book, anime/manga, comic, cartoon, and “misc”) totaling to approximately 5 million records.

The original features are:

* section
* subsection
* story (url)
* author (url)
* story\_id
* author\_id
* language
* category
* rated (e.g. "T" for Teens)
* chapter\_count
* review\_count
* word\_count
* fav\_count
* follow\_count
* published (unix timestamp)
* updated (unix timestamp)
* description (summary of the story)

One of the challenges I ran into while handling the dataset was the fact that it wasn’t completely scraped, so it’s missing a large chunk of data. However, the rest of the data is representative.

Before continuing, it’s pertinent to go over the specific jargon used by the archival site, which persists in the features of the dataset. For the purposes of this report, note that **“Fav” and “fav\_count”** refer to a specific type of engagement by readers, **similar to the “like” function** on Facebook or Twitter—a one-time click expressing appreciation of the content. **“Follows” and “follow\_count**” mean that a reader is subscribed to the work and **gets notifications when the work updates**. **“Reviews” and “review\_count”** refer to **comments left on the work**—multiple reviews can be left by the same reader, but the author cannot reply.

1. **Cleaning and Preprocessing**

This was the most time-consuming portion of the project, but infinitely the most useful, as I got to know the dataset fairly intimately and did some data exploration along the way, understanding what constituted an outlier in the dataset and what didn’t. One of the first things I did was consolidate all of the data into one .csv file for a holistic approach to the analysis process.

I set the file up in a dataframe, separated it with a tab delimiter, told the reader to ignore bad lines (these were negligible—out of a 5 million row dataset, it skipped 1000 lines) and finally, set the header to 0. The next step in the cleaning was to identify the only duplicates that should exist—since fanfiction inherently is a unique process, there would be no relevant duplicates in the metadata, and all duplicates were just the section headers that got compiled together when I consolidated the data. Each file had its own section header leading to a total of approximately 8,000 duplicates of section headers. However, because I wanted to keep the header, I set “keep = ‘first’” when dropping the duplicate rows. A screen shot of a person

Description automatically generated

After I set up the initial cleaning, I used a subset of the data in order to reduce the memory load while preprocessing the rest of the data. A few of the columns had numbers in the thousands, which had commas in them due to how the data is formatted on the source site, which I removed. I only had to do this for the following three columns, since they were the only ones throwing errors.

A black sign with white text

Description automatically generated

Because of the nature of the data, consolidation transformed all of the columns to objects/strings rather than integers (which is why str.replace worked above), and I needed those columns to be classified as integers for my conditional removal of spam and outliers. To transform the numeric columns back into integers, I used “pd.to\_numeric” and coerced errors—there were already NaN values where there should have been 0s (this is due to the fact that the source site itself does not actually *list* a 0 when there are no favs, follows, or reviews), so this led naturally to replacing all null values with 0. Then I converted all numbers to integers, just to be safe.

A screenshot of a computer

Description automatically generated

After this, I needed to remove spam and outliers. Spam is a difficult thing to measure usually without actually looking at the description, but because fanfiction tends to be a certain length in terms of chapters and word count, I could easily filter out a majority of the spam. Regarding outliers, I found that an entire separate project could be done just focusing on the outliers, but that it was not going to be pertinent for my purpose, which was finding the ideal word and chapter count for the most engagement. Generally, most fanfiction writers do not write more than 100,000 words—a lot of published novels don’t even reach this length—and chapters that are genuine and not “short story collections” are below a count of 50. Likewise, on the engagement level, certain categories only had one or two data points that were far outside the rest of the dataset, and thus were removed for the purpose of a cohesive analysis. I ended up testing this part most, especially after developing the algorithm and noticing how the results were skewed when taking the far outliers into account, which is what convinced me to create these parameters.

A green sign with white text

Description automatically generated

Finally, I set up sampling in anticipation of the algorithm. There are about 700,000 rows of data in the subset I was using (fanfiction from the Harry Potter media category), so I knew that even there, I needed to be using sampling methods. I ended up using bootstrapping, selecting approximately 10,000 for the subset and 50,000 for the entire dataset and setting the replacement to true. For the purposes of the analysis, I ended up using the entire dataset. In both cases, the output looked valid and demonstrated some interesting patterns.

A close up of a logo

Description automatically generated

1. **Data Reduction and Projection**

I reduced the number of features from 17 to 12 during preprocessing, but as may have been extrapolated from the code I showed above, the features I actually ended up using in my analysis were “word\_count”, “chapter\_count”, “fav\_count”, “follow\_count”, and “review\_count” due to the nature of the project. I wanted to keep this project at a two-dimensional level, which would allow for a more clean analysis. However, for the purposes of potentially testing a three-dimensional clustering, I kept the feature reduction as follows:

A picture containing object, device

Description automatically generated

Essentially, I dropped “story” and “author”, but kept their numerical counterparts, “story\_id” and “author\_id” due to the fact that they became redundant, and from a statistical standpoint, the numbers are more useful. For a qualitative analysis, those features would become more important, but for a quantitative analysis, they only make the process slower. Similarly, I dropped “description” because it would be more useful for a qualitative analysis or a text analysis project, which is a different project entirely. The “published” and “updated” columns would likewise be useful on a timeline trajectory project, and might be useful if I continue to iterate on this project in the future. I kept the rest in case I needed to call specific records to identify outliers and figure out if they were statistically relevant.

1. **Data Mining Task**

At the proposal level, because I have an innate understanding of the data with which I was working, I knew this was a clustering problem—there isn’t really anything supervised about fanfiction metadata, especially when it comes to engagement. I knew it would mostly give me individual data points that would be all over the map on a positive Cartesian plane, when visualized. These concepts taken together eliminated the possibility of a classification model, and instead, the problem of understanding where engagement hit its peak would be most efficiently worked out with cluster centroids, especially considering the amount of data.

1. **Algorithm**

When it came to writing the algorithm, I wanted something simple that could produce multiple clustering graphs. Because a lot of the insights can be gained by cross-referencing these graphs, I wanted something that could run relatively smoothly. Because of the millions of values I was working with, I knew that regular k-means might not be prudent in this case, so I decided to design the algorithm to run as quickly as it could. Combined with the bootstrapping I constructed in the pre-processing stage, I used mini batch k-means from the scikit-learn library, initialized it with k-means++, and set the batch\_size to 1000.

The reason I chose 10 clusters is because the sampling is still a lot of data points, and in the end, the clusters were much more clearly defined. I tested out other cluster numbers from 6-12, but they were either too cramped or too wide, and rendered the visualization meaningless.

A screen shot of a computer

Description automatically generated

1. **Data Mining & Interpretation**

I created six clustering graphs total in order to get an overall picture of what engagement looks like. Three of the graphs measure word count against the three types of engagement: favs, follows, and reviews; the other three measure chapter count against the same types of engagement.

The first graph we will be looking at measures the number of words written against the amount of reviews, and as a reminder, the word count has been limited to 100,000 and the review count has been limited to 1000 to remove outliers—beyond these numbers, the data points skew the results unevenly.

A screenshot of a cell phone

Description automatically generated

The patterns in the above graph show that as the word count increases, review count also tends to increase rather drastically, but seems to reach its peak around the 60k-80k cluster. The clusters appear to grow bigger as the word count gets higher, with far more data points scattering outwards once we reach approximately 300 reviews. It’s also of note that most of the cluster centroids (five, to be exact) are focused in around the 0-20,000 area, simply because it’s much easier to write a shorter piece than it is to devote more time and energy. However, clearly the effort pays off in the form of comments—though interestingly, “the more you write, the more people will comment” does not hold true throughout, as overall, readers will comment less on works above 80,000 words. This analysis should be taken with a grain of salt, as this does not take into account the quality of a work, but clearly there is a pattern of more commentary at higher word counts.

A screenshot of a cell phone

Description automatically generated

On a similar level, the likelihood of a reader to follow a piece of fanfiction (as a reminder, this means the reader receives notifications that the piece has updated with a new chapter) increases with the word count. Pieces of fanfiction that don’t have as high of a word count are logically less likely to have multiple updates, and so are less likely to have a person subscribe, which is why we see a slight lift in the bottom-most cluster as the follow count increases. There is a similar pattern here to the review count graph, but it’s definitely less drastic.

A screenshot of a cell phone

Description automatically generated

The data analysis largely holds true with regard to favs. What’s surprising about this graph in comparison to the rest is the density is much more concentrated in the lower word count tiers as well as the fact that pieces that are more likely to get more favs are in the 80k-100k range instead of the previously seen 60-80k range. Additionally, the lowest cluster extends much farther outwards than both the reviews graph and the follows graph, seeming to indicate that people are more likely to fav a work at a lower word count. The centroids of the lower clusters being more concentrated near the origin of the graph most likely indicates that more of the records in that range are closer to the 0 point on the x-axis, but visually, it’s fairly clear to see that despite the distribution, a fic that has a lower amount of words can still get a relatively high amount of favs, unlike follows or reviews.

From here, we’ll look at chapter-based analytics, which are vastly different from the word count-based clusters in that they cluster vertically as opposed to horizontally.

A close up of a piece of paper

Description automatically generated

Interestingly, the clusters here are not as close together as they were in the word count graphs, and you can plainly see where there are gaps in the clusters, especially around the 200-300 review count cluster, where the chapter counts are five or higher. Moreover, reviews tend to come about in higher amounts for fics that have 10-30 chapters, which makes sense given that a reader can leave multiple reviews—and in actuality, can leave a review *per chapter*. What’s most surprising about this, however, is the fact that there seems to be a “sweet spot” of reviews—the 200-300 review cluster is one of the biggest and seemingly most dense clusters. This brings the centroid a little lower than the others in terms of the natural curve they seem to show, which means that in theory, someone could post a fic with 15 chapters and still receive 200-300 reviews total.

A close up of a map

Description automatically generated

In the chapters vs. follows graph, an incredible thing happens, which is that one of the centroids breaks free of the previously clearly stratified space—we actually have three separate clusters around the 0-15 follows: one in purple that seems to show 0 follows for anywhere from 0-15 chapters, one in green that seems to show 1-10 follows for the same amount, and then that huge yellow cluster of anywhere from 0-20 follows for 15-50 chapters. This seems to show that there is a high density of records in the yellow cluster, which is why it wasn’t as stratified. Theoretically, the separation could indicate that the author published all of the chapters at once, which would account for someone not wanting to follow the story, or it could indicate that the story has been complete for some time, and that the reader has removed the story from their follow list.

In general, a fascinating thing about chapters vs. follows is that a higher amount of chapters and a higher amount of follows naturally indicates that an author has been regularly updating the work, and updating the work pushes the work to the top of the archive, thereby causing a chain-reaction effect in which works with more chapters will naturally obtain more engagement, which is likely why the data from the chapters vs. reviews graph looked the way that it did. This is not to say that this is the case in general, but it is definitely one explanation for the way the centroids end up curving.

A screenshot of a cell phone

Description automatically generated

The same pattern that we see in the follows graph happens in the favs graph too, which seems to indicate that favs/follows have more in common with each other than they do with reviews. The two are not intrinsically tied in the site’s construction, but the correlation exists, and might indicate that someone who is likely to follow a fic is also likely to fav it. There is also something unique in this graph, which is that fics with only one chapter can get any amount of favs, creating a visual floor on this particular graph, unlike the other graphs. Another interesting thing about this graph as opposed to the other two is the density of the centroids—where we saw a much more even spacing of the centroids in the chapters vs. reviews graph, there is more closeness between most of the centroids in this graph, which might indicate that favs are not as freely given as reviews. Overall, however, that might be better for learning—after all, a review can give specific feedback that a fav just can’t do with a press of a button.

**Conclusion**

When it comes to the amount of writing someone does when they write fic and how much attention they get for it, it’s natural to assume that the more someone writes, the more attention they will get. However, as the data has shown, the reality is more complicated than that. There does indeed appear to be a “sweet spot” for how big a work is and how much engagement it gets, and on a median level, taking all levels of engagement into account, that sweet spot appears to be 60,000 to 80,000 words and 10-20 chapters. Though most fics will not see a huge amount of engagement, it stands to reason that any burgeoning writer looking to strengthen their writing skills can definitely write a transformative work about their favorite piece of media and, as long as they put the time and effort into it, they can get a solid amount of feedback on their work, which can serve to make them stronger writers on the whole.