Venkata Praveen Wunnava. Exploration of Wikipedia traffic data to analyze the relationship between multiple pages. A Master's Paper for the M.S. in I.S degree. May, 2020. 44 pages. Advisor: Jaime Arguello

Time series analysis and forecasting is an essential part of any holistic data analysis. Many prediction challenges depend on correctly assessing how the data changes over time. Several examples of time-series analyses that are traditionally seen are of univariate type, but it is hardly the case in a real-world setting. Any time series is influenced by multiple components, which includes its past and other constant or variable factors. This project is aimed at understanding multivariate time series models using the non-traditional time series analysis like clustering and sequence to sequence model using a long short-term memory architecture. The dataset on which this experiment is being applied is the Wikipedia web page traffic. The dataset contains around 145000 web pages and corresponding web page traffic from July 2015 to December 2016. Findings based on the hierarchical clustering model are presented in this study.

Headings:

Time series analysis

Multi-step time series forecasting

Hierarchical Clustering

Dynamic time warping

Data Analysis

Sequence to Sequence model

LSTM model

Wikipedia Traffic pages

EXPLORATION OF WIKIPEDIA TRAFFIC DATA TO ANALYZE THE RELATIONSHIP BETWEEN MULTIPLE PAGES

by
Venkata Praveen Wunnava

A Master's paper submitted to the faculty
of the School of Information and Library Science
of the University of North Carolina at Chapel Hill
in partial fulfillment of the requirements
for the degree of Master of Science in
Information Science.

Chapel Hill, North Carolina

May 2020

Approved by

Jaime Arguello

Table of Contents

BACKGROUND AND MOTIVATION

The practice of understanding how data changes over time and to make predictions based on that knowledge has been prevalent almost since the beginning. Before we humans did it using our intuition, for example, to predict the weather in a geographical region and grow certain crops accordingly. But now, with all the computation power and the availability of data, this has become even more ubiquitous in any comprehensive, in-depth data analysis. Applications of time series range from a) econometrics to understand and forecast a deal outcome b) in stock markets to predict the stock market movements c) behavioral science to understand how various factors affect mood at a certain point in time (Stock, 2001) d) others include heights of ocean tides, count of sunspots (Wikipedia, n.d.). Even with such a wide range of applications, a fundamental univariate time series analysis is a complex process. It involves decomposing a time series to extract relevant statistics and characteristics like moving average or regression. A typical decomposition an additive time series looks like:

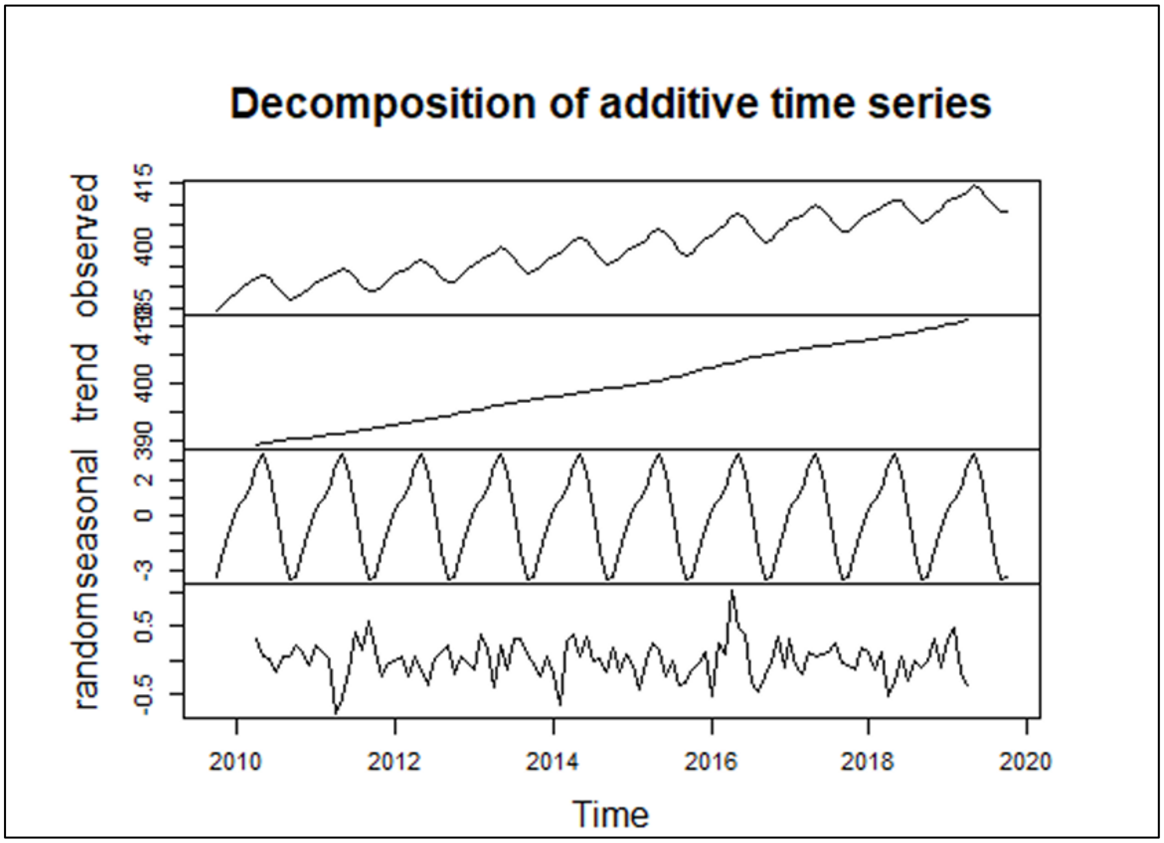


Figure 1: Time series decomposition

The above plot shows the CO2 Level at Mauna Loa (CO_2 and Greenhouse Gas Emissions, n.d.) measures across several consecutive years. As can be seen, the first box shows the observed Co2 values, the second box shows the trend, the third box shows seasonality, and the last one is just some random noise that cannot be predicted. In order to forecast values, one must make use of such decomposition and predict the trends and seasonality separately or together. Such a process leads to creating multiple statistical models, which makes it complicated for the general audience and a few data analysts alike. This is true even for other approaches to time series analysis like frequency-based or Fast Fourier Transformation. The complexity of such a time series compounds when the analysis extends to multivariate time series. The above series is univariate as the CO2 level is dependent on a single variable, which is time. But add to this, factors such as temperature, humidity, wind speed, or other external factors, which is relevant in a real-world scenario, the analysis would need to fine-tune further. Multivariate time series analysis uses techniques like Vector Autoregressive processes, which is an extension of the univariate autoregression model (Stock, 2001) or uses machine learning techniques like neural networks and SVM.

Additionally, anytime series forecasting involves a single step prediction that predicts value at t+1. And a multi-step prediction, which predicts the value at 't+n,' where 'n' depends on the model and the requirement. As a part of this project, I have implemented a multivariate time series model that has a multi-step prediction.

In the case of time series that are non-volatile, the analysis and forecasting using the traditional techniques that help understand trend and seasonality should suffice (Kotu & Deshpande, 2015). But in other cases, which is more frequent, it helps to create a more generalizable and dynamic technique that could factor in multiple inputs and outputs. The downside to making a generalizable model is that It becomes very brittle and inflexible (Taylor & Letham, n.d.). Moreover, in today's scenario where information is produced at petabytes per second (DATA NEVER SLEEPS 6.0, n.d.), analysis and forecasting are needed at a much faster rate than before. Also, since time series forecasting is still a niche subject, especially in the field of data analytics and data science, any high-quality forecasting requires considerable experience and manual effort. Looking at the above concerns, features like faster analysis, more expertise, transferable, and robust models are need of the hour in time series forecasting.

There is a certain amount of work that is being done in this field, for example, fakebooks' open-source Prophet, a forecasting tool that is available in R and Python. Prophet was conceived at Facebook and is optimized for the business forecast tasks encountered at Facebook (Taylor & Letham, n.d.). The primary motivation of this study is to explore techniques that are generalizable, transferable, and give an excellent time series related insight to an audience that is not well-versed with traditional time series analysis. Even though I have created the study, I consider myself a suitable audience as well, because, to begin with, I didn't have much background in time series analysis. Through this study, I aimed to understand the underlying intricacies of such analyses. And put in a fashion which is easily understood by an audience with limited understanding of time series analysis.

METHOD

1.1 Data Collection

The dataset for this project is taken from Kaggle. The dataset was produced three years ago as part of a competition that forecasts the future web traffic for approximately 145,000 Wikipedia articles. The data is originally provided by Google Inc. in a simple two-dimensional matrix, which has dates as headers and values of page visits against each Wikipedia page. In essence, there are 145,000-time series, which is represented by each row. The dataset doesn't have a lot of information about the page except the name of the page, access type, and agent through which the webpage is accessed. The categories of type of "access" and "agent" are below:

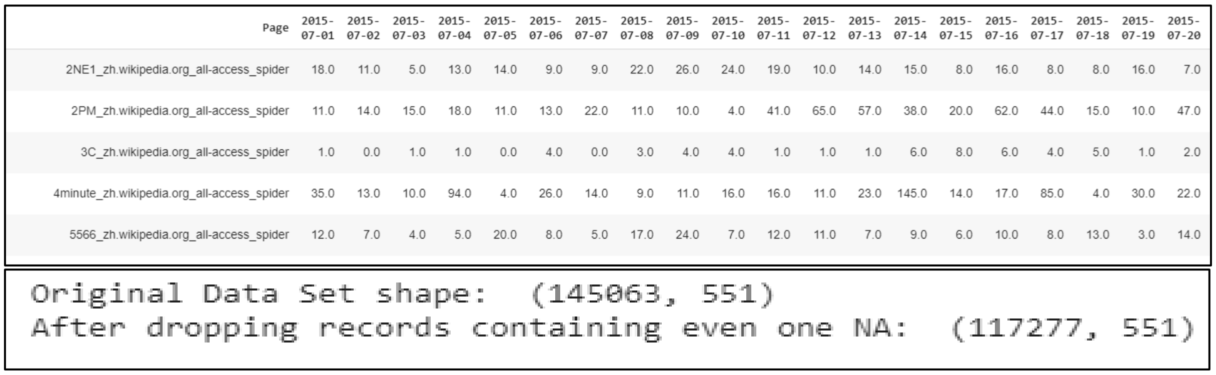
Types of access: Desktop, Mobile, All-access

Types of agent: Spider, All agents

The spider agent above refers to a web crawler. "*A Web crawler, sometimes called a spider or spiderbot and often shortened to crawler, is an Internet bot that systematically browses the World Wide Web, typically for the purpose of Web indexing" (web spidering)* (Wikipedia, n.d.)

1.2 Exploratory Data Analysis

As mentioned earlier, the data set contains 145000-time series, which is more than required for this study. The below picture gives a snapshot of data and the shape of data after removing records that have no values attached to them (for any one day).

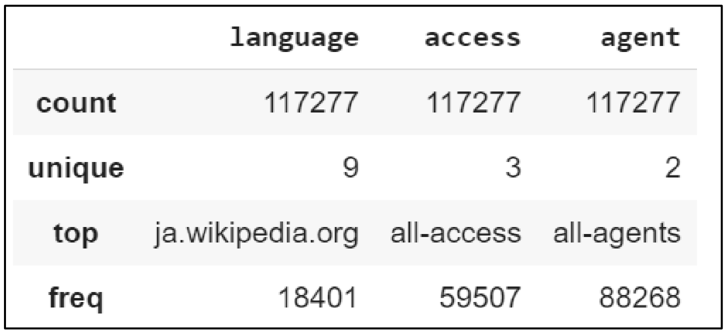


Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10	2015-07-11	2015-07-12	2015-07-13	2015-07-14	2015-07-15	2015-07-16	2015-07-17	2015-07-18	2015-07-19	2015-07-20
2NE1_zh.wikipedia.org_all-access_spider	18.0	11.0	5.0	13.0	14.0	9.0	9.0	22.0	26.0	24.0	19.0	10.0	14.0	15.0	8.0	16.0	8.0	8.0	16.0	7.0
2PM_zh.wikipedia.org_all-access_spider	11.0	14.0	15.0	18.0	11.0	13.0	22.0	11.0	10.0	4.0	41.0	65.0	57.0	38.0	20.0	62.0	44.0	15.0	10.0	47.0
3C_zh.wikipedia.org_all-access_spider	1.0	0.0	1.0	1.0	0.0	4.0	0.0	3.0	4.0	4.0	1.0	1.0	1.0	6.0	8.0	6.0	4.0	5.0	1.0	2.0
4minute_zh.wikipedia.org_all-access_spider	35.0	13.0	10.0	94.0	4.0	26.0	14.0	9.0	11.0	16.0	16.0	11.0	23.0	145.0	14.0	17.0	85.0	4.0	30.0	22.0
5566_zh.wikipedia.org_all-access_spider	12.0	7.0	4.0	5.0	20.0	8.0	5.0	17.0	24.0	7.0	12.0	11.0	7.0	9.0	6.0	10.0	8.0	13.0	3.0	14.0

```
Original Data Set shape:  (145063, 551)
After dropping records containing even one NA:  (117277, 551)
```

Figure 2: Data Snapshot

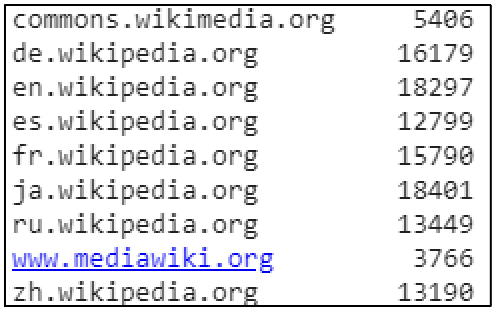
Further analysis of data revealed that the dataset contains pages from multiple languages like English, Japanese, French, and other languages. A frequency count of pages again access type and agents look like:



	language	access	agent
count	117277	117277	117277
unique	9	3	2
top	ja.wikipedia.org	all-access	all-agents
freq	18401	59507	88268

Figure 3: Dataset Description

The top row depicts the numbers of webpages that have complete data (Kaggle, n.d.). The bottom two rows give the detail on the most frequently occurring page, access, and agent. These will help to analyze how and which pages are accessed by the users. The type of access and agent are already discussed in the earlier section. The type of project (or language) can be drilled down further into



```
commons.wikimedia.org      5406
de.wikipedia.org          16179
en.wikipedia.org          18297
es.wikipedia.org          12799
fr.wikipedia.org          15790
ja.wikipedia.org          18401
ru.wikipedia.org          13449
www.mediawiki.org          3766
zh.wikipedia.org          13190
```

Figure 4: Language Summary

A sorted bar chart of the same is below:

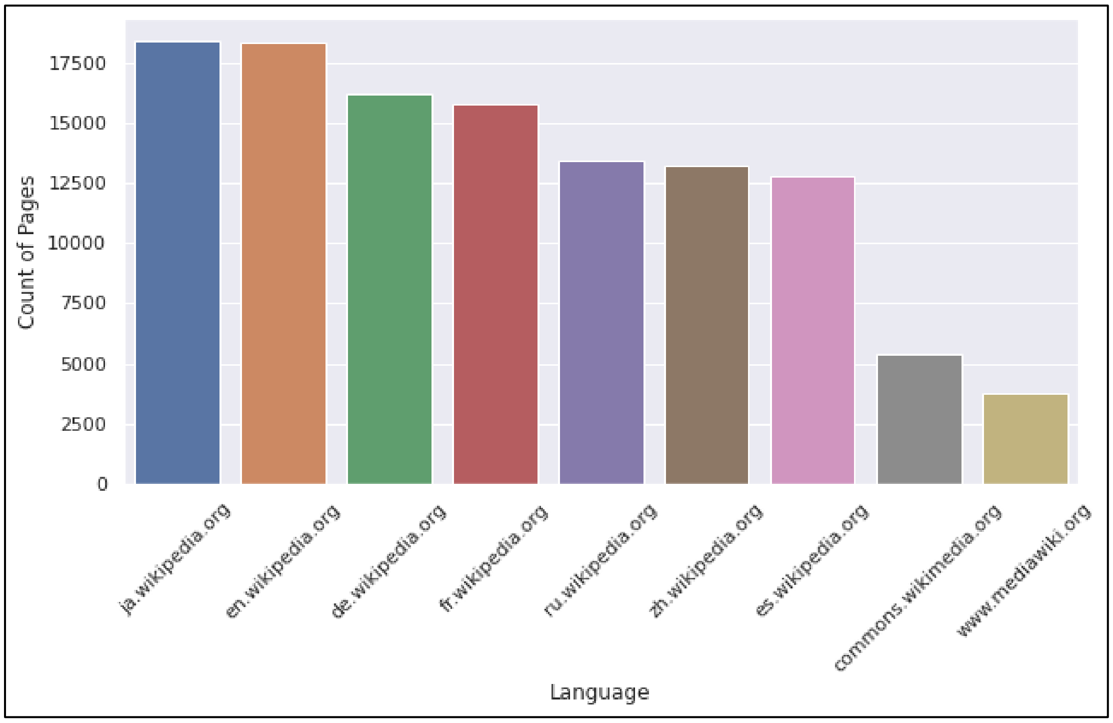


Figure 5: Language Summary Plot

Even though the data shows the highest amount of "ja" (Japan) domain webpages, further analysis showed that the English Wikipedia page has the most visitors.

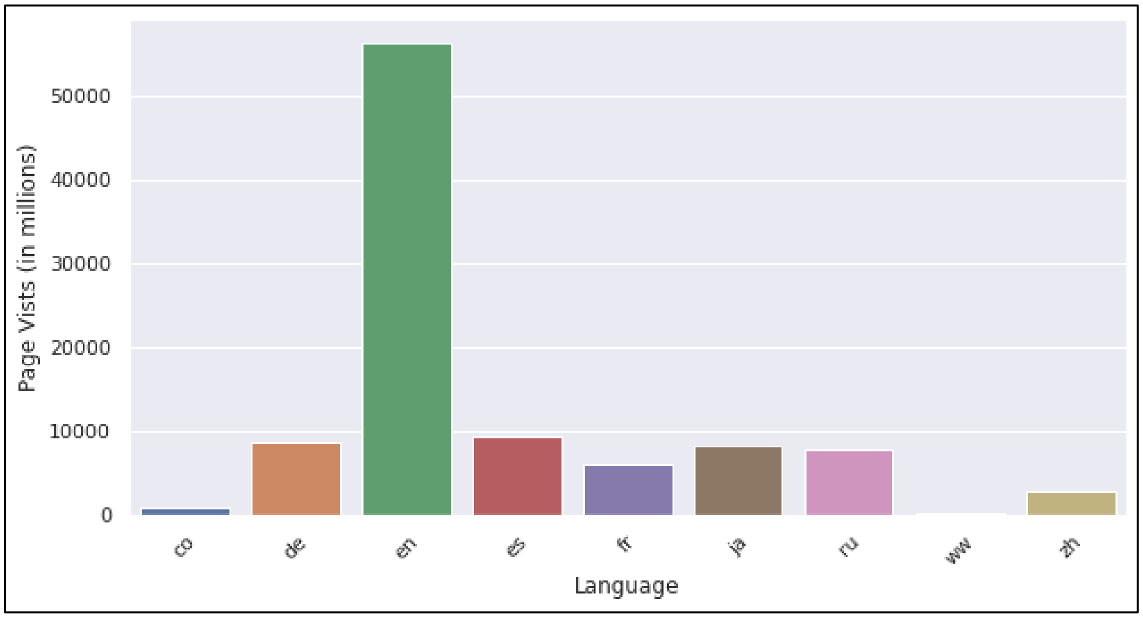
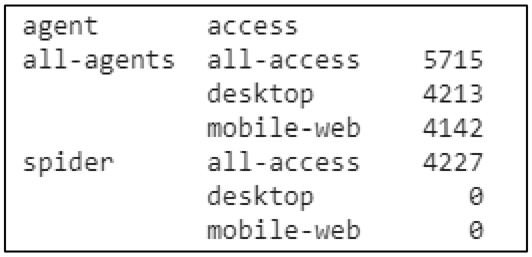


Figure 5: Language Summary Plot

The purpose of this study is to understand how different time series are associated, judged by how humans interact with multiple pages, i.e., not by spiderbots but by humans. But eliminating any sort spiderbots, agent-type is done only on the "English" Wikipedia pages rather than all Wikipedia pages. The idea is that a user would navigate across wiki pages only in one language. Such an assumption is safe because it is highly unlikely that a user accesses one page in English and another page of a related topic in French or any other language. Thus, further filtering the dataset to take only the pages from 'en.wikipedia.org' project and grouping the agents and access type reveals:



```
agent       access
all-agents  all-access    5715
            desktop       4213
            mobile-web    4142
spider      all-access    4227
            desktop          0
            mobile-web       0
```

Figure 6: English Wiki Pages grouped by agent and access

A bar chart of the same below:

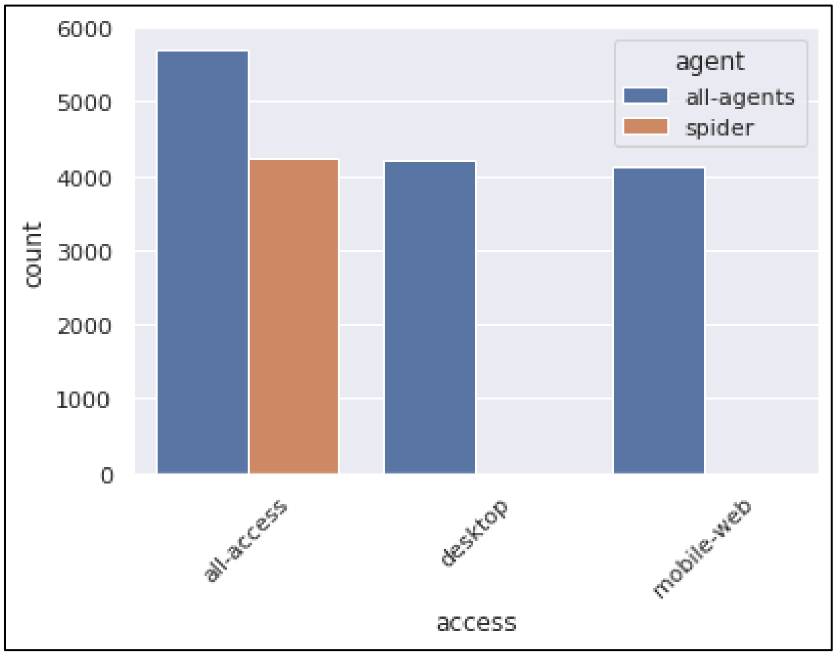
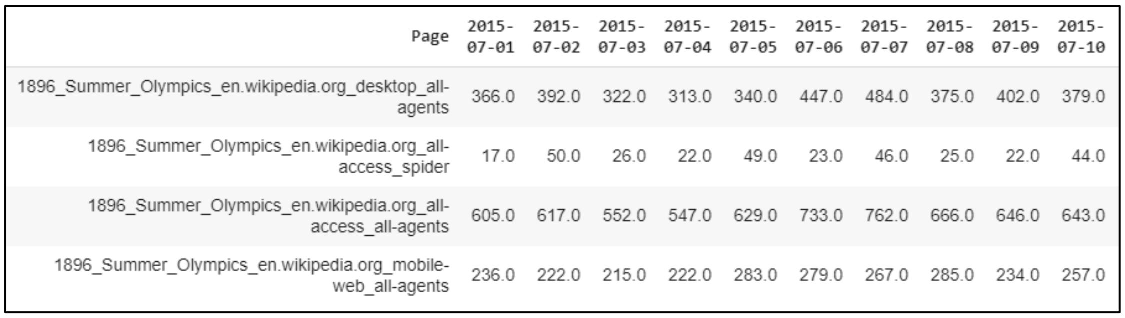


Figure 7: Plot of English Wiki Pages grouped by agent and access

To give an idea of what each of the bar in Figure 7 entails, refer to the below snapshot of the Wikipedia pages with the topic "1896 Summer Olympics"

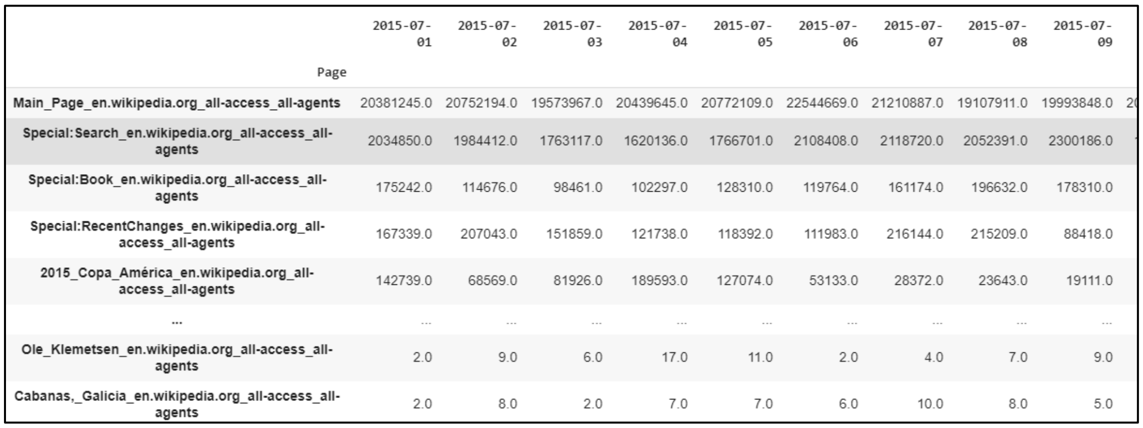


Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10
1896_Summer_Olympics_en.wikipedia.org_desktop_all-agents	366.0	392.0	322.0	313.0	340.0	447.0	484.0	375.0	402.0	379.0
1896_Summer_Olympics_en.wikipedia.org_all-access_spider	17.0	50.0	26.0	22.0	49.0	23.0	46.0	25.0	22.0	44.0
1896_Summer_Olympics_en.wikipedia.org_all-access_all-agents	605.0	617.0	552.0	547.0	629.0	733.0	762.0	666.0	646.0	643.0
1896_Summer_Olympics_en.wikipedia.org_mobile-web_all-agents	236.0	222.0	215.0	222.0	283.0	279.0	267.0	285.0	234.0	257.0

Figure 8: Snapshot of records associated with a single topic

Each record indicates how the number of visitations is captured of the pages with the same topic but different access types and agents. For the purpose of this study, I decided to take pages that contain figures for all-access and all-agents. There is a possibility that all-agents might include data from spider agents. Still, such a theory could not be eliminated since the number of page visitations by "all-access and all-agents" (2nd last row above) is unequal to the sum of all the other rows for a particular day. So, to be all-inclusive and potentially not loose relevant codependent data, the pages which have all-access and all-agents properties were taken into the study.

Finally, some of the most common but irrelevant (in terms of this study) pages like Wikipedia home page and special pages (search, changes, and books) that could highly skew the data (towards the extreme right end) are deleted.



Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09
Main_Page_en.wikipedia.org_all-access_all-agents	20381245.0	20752194.0	19573967.0	20439645.0	20772109.0	22544669.0	21210887.0	19107911.0	19993848.0
Special:Search_en.wikipedia.org_all-access_all-agents	2034850.0	1984412.0	1763117.0	1620136.0	1766701.0	2108408.0	2118720.0	2052391.0	2300186.0
Special:Book_en.wikipedia.org_all-access_all-agents	175242.0	114676.0	98461.0	102297.0	128310.0	119764.0	161174.0	196632.0	178310.0
Special:RecentChanges_en.wikipedia.org_all-access_all-agents	167339.0	207043.0	151859.0	121738.0	118392.0	111983.0	216144.0	215209.0	88418.0
2015_Copa_América_en.wikipedia.org_all-access_all-agents	142739.0	68569.0	81926.0	189593.0	127074.0	53133.0	28372.0	23643.0	19111.0
...	...	...	...	...	...	...	...	...	...
Ole_Klemetsen_en.wikipedia.org_all-access_all-agents	2.0	9.0	6.0	17.0	11.0	2.0	4.0	7.0	9.0
Cabanas,_Galicia_en.wikipedia.org_all-access_all-agents	2.0	8.0	2.0	7.0	7.0	6.0	10.0	8.0	5.0

Figure 9: Snapshot of the top four visited pages

The top four records in the above figure are removed. The final dataset count is 5711 time series over 550 days spanning from 1st July 2015 to 31st December 2016.

The final plot of time series looks like:

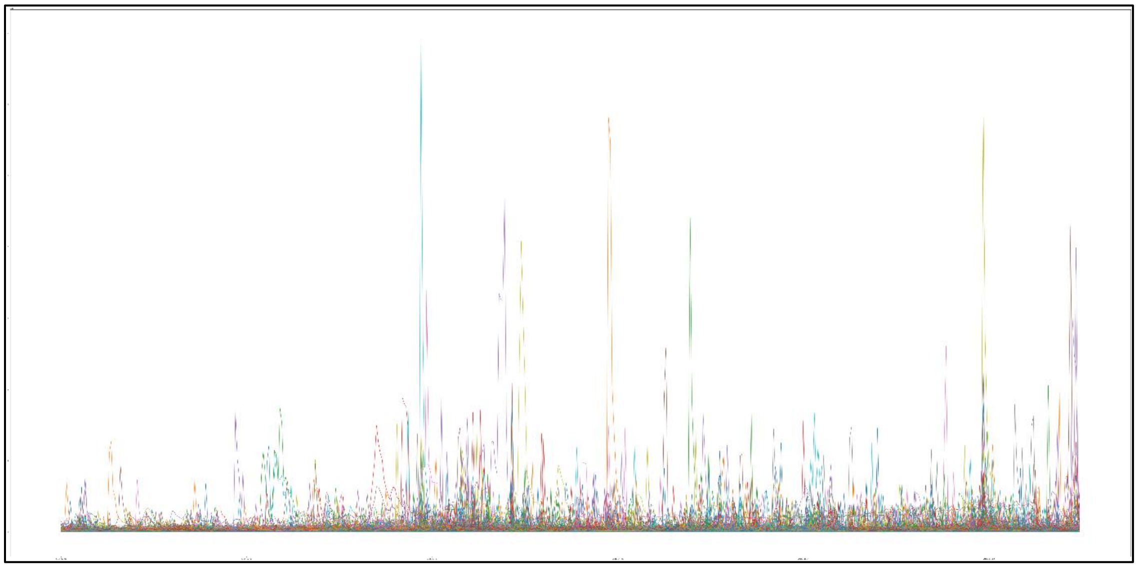


Figure 10: Plot of all the time series in the final dataset

A more condensed plot resulting from taking Log (#Views)

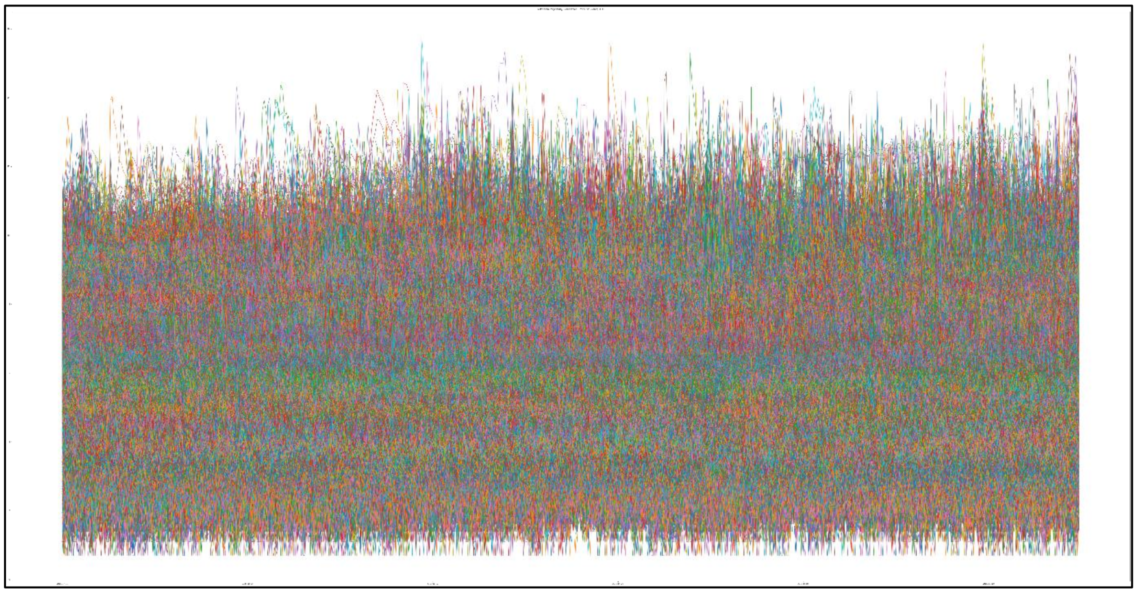


Figure 11: Plot after taking the Log of the number of views

As can be seen, there are a varying number of spikes in specific time-periods. Not all series spike at the same time, some pages follow the spike caused by other pages, thus indicating causality between the Wikipedia pages.

1.3 General Correlation:

As part of this study, I will be making use of two different correlation metrics to infer the similarity between temporal data. A bit of background on the similarity measures that will be used in this paper. Below is a figure that depicts the difference between Euclidean and DTW time series similarity measurement (Cassisi, 2012):

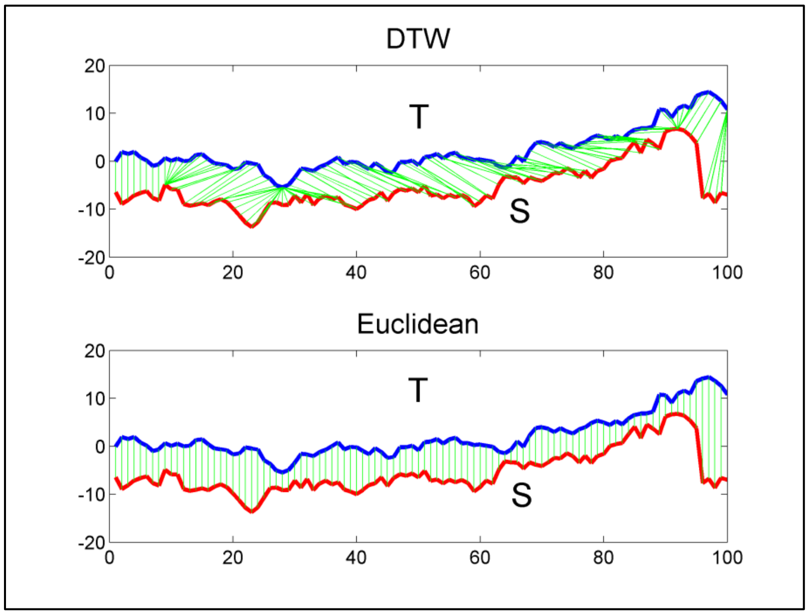


Figure 12: Difference between DTW and Euclidean distance (Cassisi, 2012)

Euclidean Distance – It is the sum of a point to point distance between two different time series, without accounting for lag or whether the series have different lengths (Cheong, 2019). As can be seen from the figure above, it is a point to point distance measurement (Cassisi, 2012).

Dynamic Time Warping (DTW) has the feature that allows for identifying patterns even with a time lag by shifting the signal (moving the time series by n steps) to compute the minimum distance between two different time series/signals. *DTW computes the Euclidean distance at each frame across every other frames to compute the minimum* *path that will match the two signals* (Cheong, 2019). Thus, DTW has many to one (or vice versa) point comparisons; making is computationally expensive.

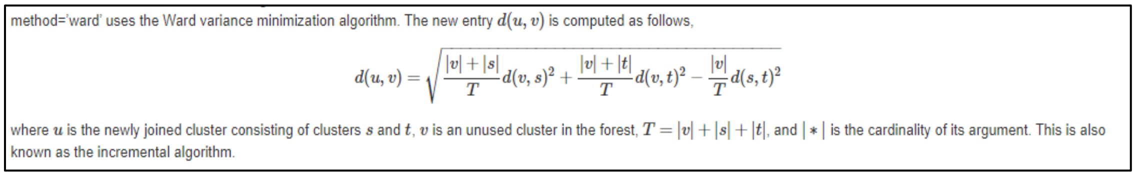
1.4 Agglomerative Hierarchical Clustering:

In this experiment, the Euclidean metric is preferred over DTW because:

1. A trending topic may not die immediately. Usually, a topic trends for weeks together (trending window), thus any semantically related Wikipedia page also will have more visitations in that trending window period. Therefore, restraining the disadvantage of a point to point comparison.
2. DTW is beneficial in time series that have unequal lengths (Cassisi, 2012), but in this study, the data set has no missing values, and all series are of equal measure.
3. Since space and time complexity of calculating the DTW distance for 5711 time-series records with 511 data points is immense without giving any potential benefit. (5710*511*511 compared to 5710*511).

DTW is an excellent technique to use to clustering the time series data, but in this scenario, it might not be the optimum and efficient approach.

The existing python SciPy library is used to create the linkages of the hierarchical clustering. There are two steps involved in this process. The first step is to find the distance between the time series, which is being done through the Euclidean metric. The next step is to cluster them based on the similarity distance. The SciPy library suggests using the Ward variance minimization algorithm. The formula for which is given by: Note that d(s,t) represents the distance between two clusters s and t (SciPy.org, n.d.).



method='ward' uses the Ward variance minimization algorithm. The new entry $d(u,v)$ is computed as follows,

$$d(u,v) = \sqrt{\frac{|v|+|s|}{T}d(v,s)^2 + \frac{|v|+|t|}{T}d(v,t)^2 - \frac{|v|}{T}d(s,t)^2}$$

where u is the newly joined cluster consisting of clusters s and t, v is an unused cluster in the forest, $T = |v| + |s| + |t|$, and $|*|$ is the cardinality of its argument. This is also known as the incremental algorithm.

Figure 13: Snapshot of Ward variance formula used (SciPy.org, n.d.)

Based on the above clustering technique, linkages are created, and a dendrogram is plotted. This dendrogram did not yield anything informative because the plot looked cluttered and confusing. After that, the data is then normalized to give a more standard range to the distance. The dendrogram is plotted again and shown below:

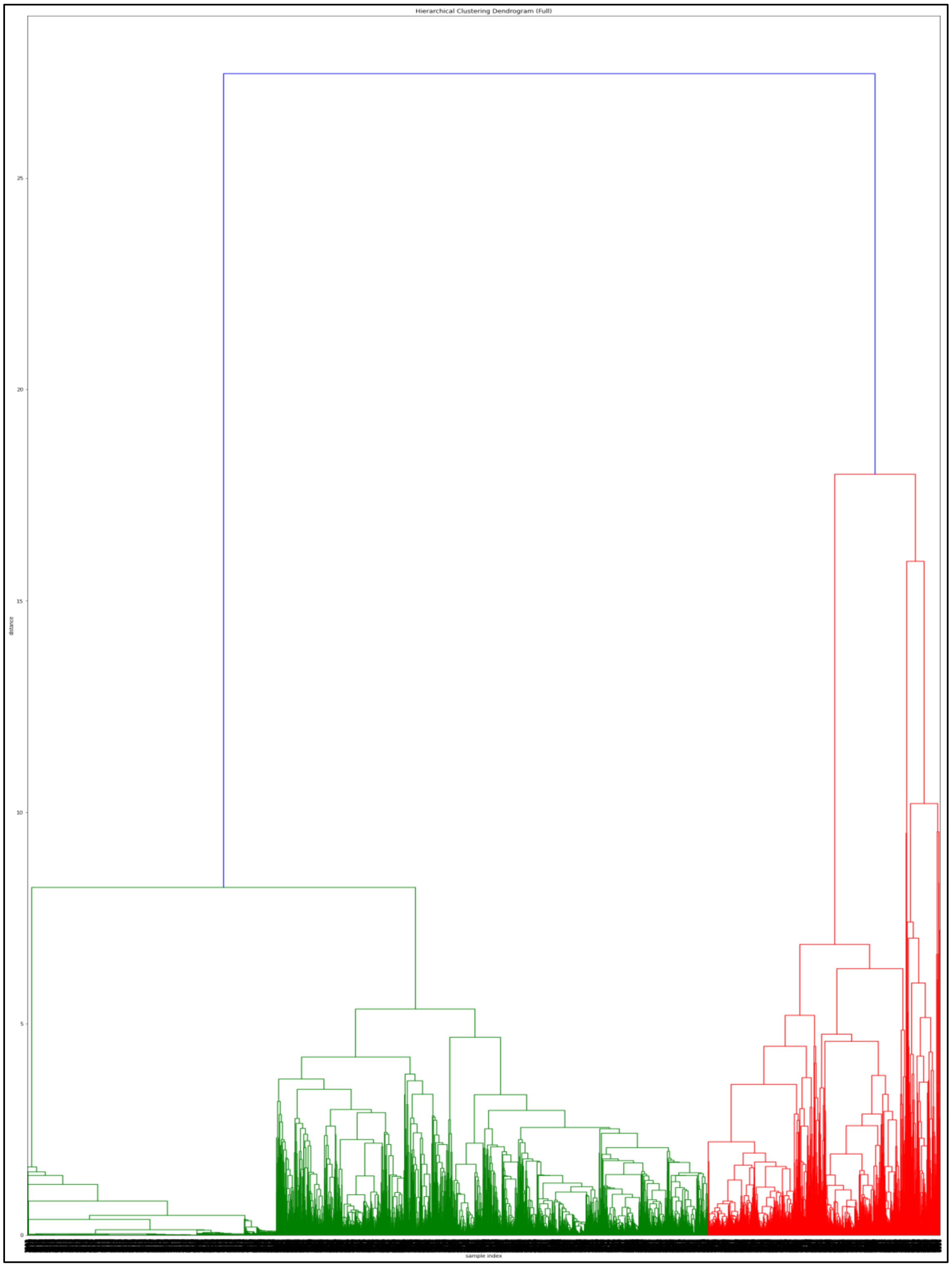


Figure 14: Plot of normalized Hierarchal Clustering

The X-Axis has the Wikipedia pages as an individual cluster, and the Y-Axis has the Ward distance. SciPy algorithm runs until only one cluster is formed. Such a bottom-up approach helps us to visualize similarity or dissimilarity between the clusters. But note the last 2 clusters (green region and red region), there does not seem to be any regular pattern. Further, since many data points appear merged, a piece of the dendrogram can be truncated and plotted as below.

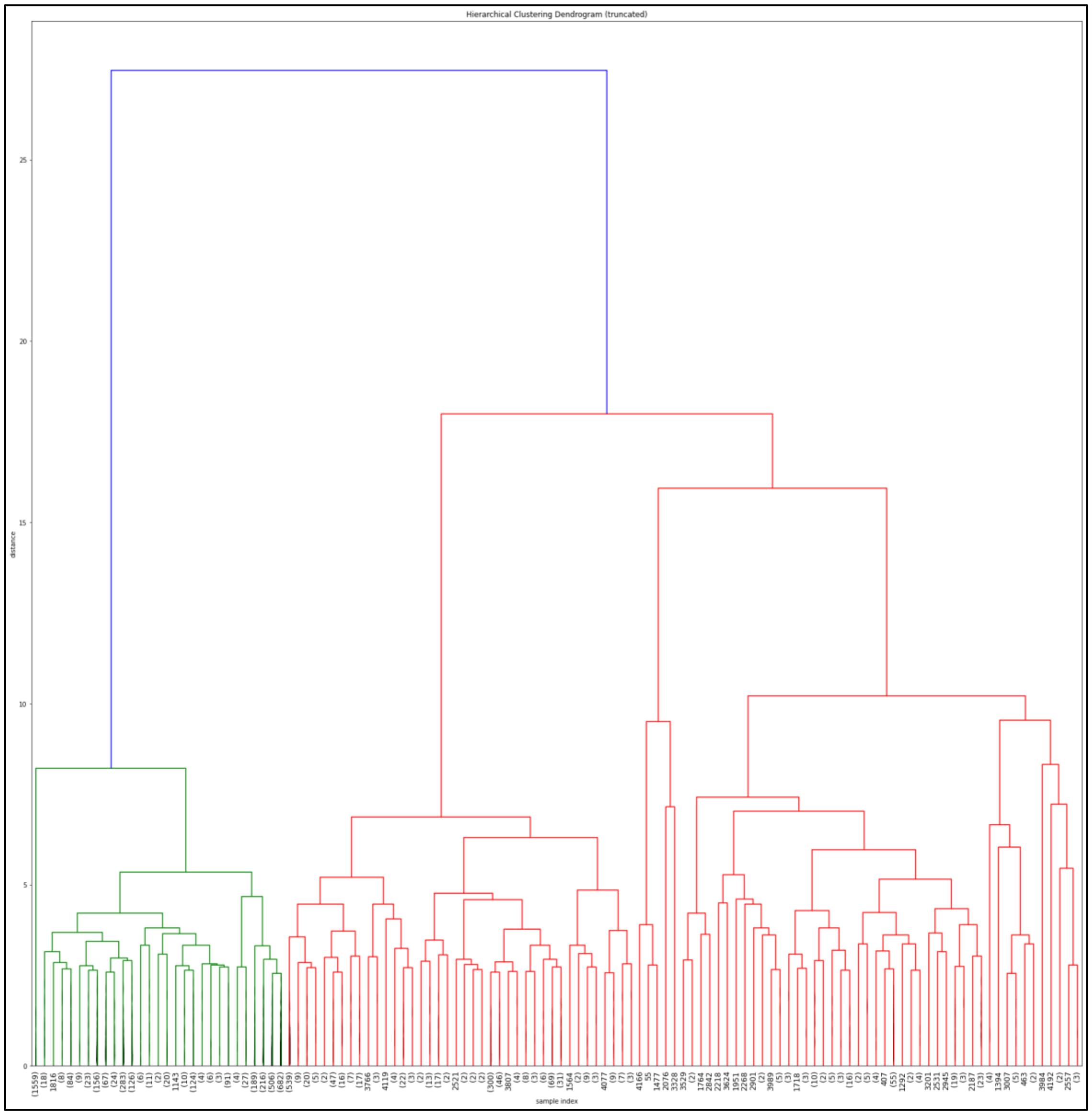
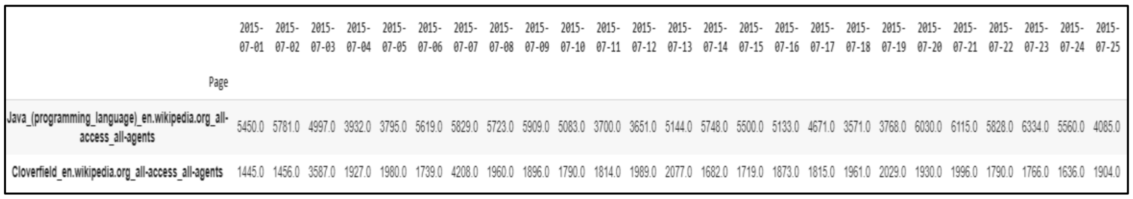


Figure 15: Plot of a truncated (last 120 clusters) normalized Hierarchal Clustering

The above dendrogram was plotted for the last 120 merges. The numbers in bracket (in X-Axis) represent the leaf counts, which implies how many samples the cluster already consisted before the last 120 merges (Hees, n.d.). Note that the number in the first bracket (from left) is 1559, which implies that the cluster already merged 1559 samples, and more are being added. To understand the clustering further, two examples from the "red area" were picked to be analyzed. The samples are chosen such that they have the maximum distance in the "red area" (i.e., samples from the extreme right and extreme left in the 2nd cluster from the top):

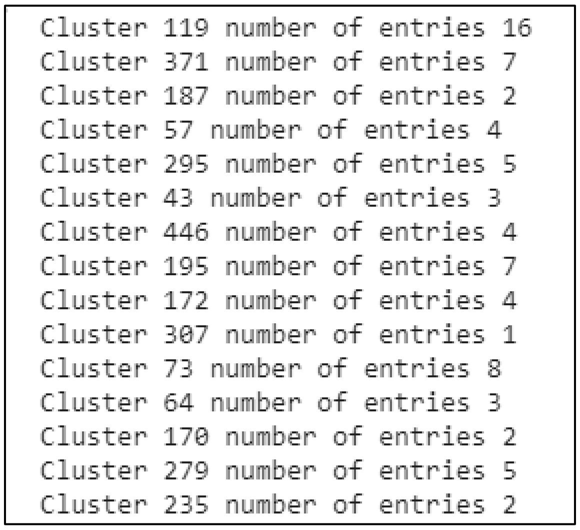


	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10	2015-07-11	2015-07-12	2015-07-13	2015-07-14	2015-07-15	2015-07-16	2015-07-17	2015-07-18	2015-07-19	2015-07-20	2015-07-21	2015-07-22	2015-07-23	2015-07-24	2015-07-25
Page																									
Java_(programming_language)_en.wikipedia.org_all-access_all-agents	5450.0	5781.0	4997.0	3932.0	3795.0	5619.0	5829.0	5723.0	5909.0	5083.0	3700.0	3651.0	5144.0	5748.0	5500.0	5133.0	4671.0	3571.0	3768.0	6030.0	6115.0	5828.0	6334.0	5560.0	4085.0
Cloverfield_en.wikipedia.org_all-access_all-agents	1445.0	1456.0	3587.0	1927.0	1980.0	1739.0	4208.0	1960.0	1896.0	1790.0	1814.0	1989.0	2077.0	1682.0	1719.0	1873.0	1815.0	1961.0	2029.0	1930.0	1996.0	1790.0	1766.0	1636.0	1904.0

Figure 16: Snapshot of records from the 2nd single cluster

As is evident from general sense and intuition, the two topics "Java" and "Cloverfield" are not related. One is a programming language, and the other is a movie. Thus, the dendrogram does not reveal any apparent clustering and is challenging to create a Ward distance cut off (that easily divides the hierarchy into discernible clusters). The first cluster ("green area") has a massive jump in Ward distance when being merged with the second cluster ("red area"), implying that they don't belong together. But such relative high jumps are seen even within the clusters lower in the hierarchy.

Thus, leading to believe that it makes sense to create a large number of clusters and look for any emerging patterns. SciPy.cluster.hierarchy.fcluster (*forms flat clusters from the hierarchical clustering defined by the given linkage matrix* (SciPy, n.d.)) can be used to flatten the dendrogram and create a requested/configured amount of clusters. In order to get an insight into the data, the library is used to create 500 clusters that will be discussed in the results sections:



```
Cluster 119 number of entries 16
Cluster 371 number of entries 7
Cluster 187 number of entries 2
Cluster 57 number of entries 4
Cluster 295 number of entries 5
Cluster 43 number of entries 3
Cluster 446 number of entries 4
Cluster 195 number of entries 7
Cluster 172 number of entries 4
Cluster 307 number of entries 1
Cluster 73 number of entries 8
Cluster 64 number of entries 3
Cluster 170 number of entries 2
Cluster 279 number of entries 5
Cluster 235 number of entries 2
```

Figure 17: Snapshot of cluster formation

RESULTS

After creating 500 clusters, clusters of five, fifteen, thirty, and the two largest clusters (of length 1555 and 281) are picked for analysis. The numbers five, fifteen, and thirty are chosen to understand the evolving pattern when moving from a low number of data points to a higher amount. The two largest clusters are selected to comprehend if any pattern appears or even an additional number of clusters should be created. The count of clusters against each number of elements is below:

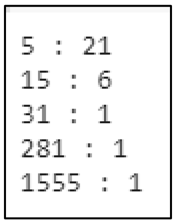
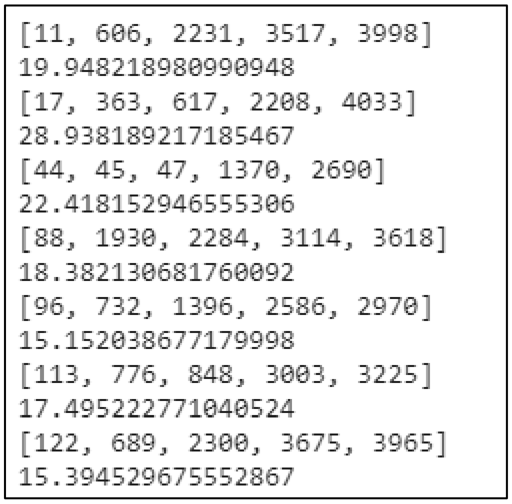

```
5 : 21
15 : 6
31 : 1
281 : 1
1555 : 1
```

Figure 18: Snapshot of selected clusters and their frequency

1.5 The cluster of five elements

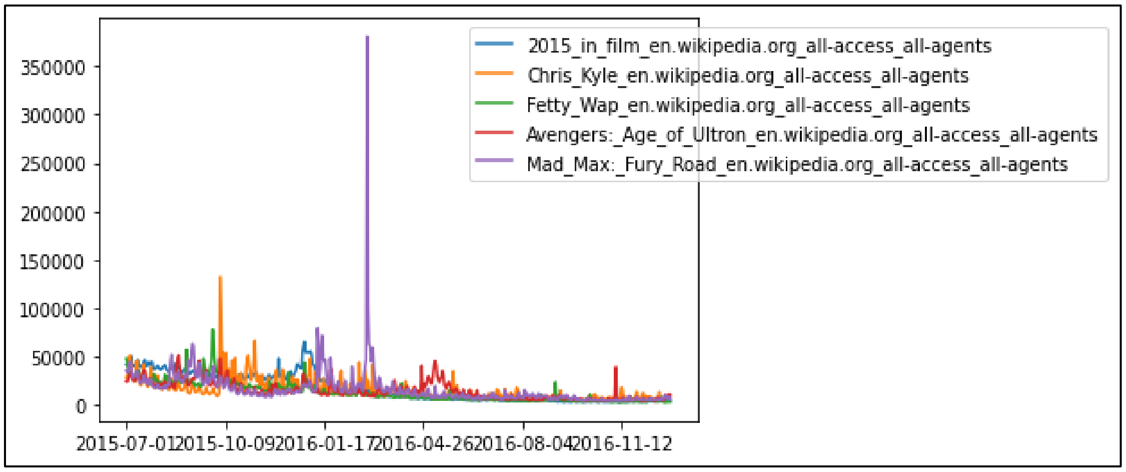
There are 21 clusters with five elements. Since it is difficult to analyze every one of them, four clusters were picked based on the highest and lowest Euclidean distance similarity score. Below is a snapshot of elements in a cluster of five and their corresponding sum of Euclidean distances.

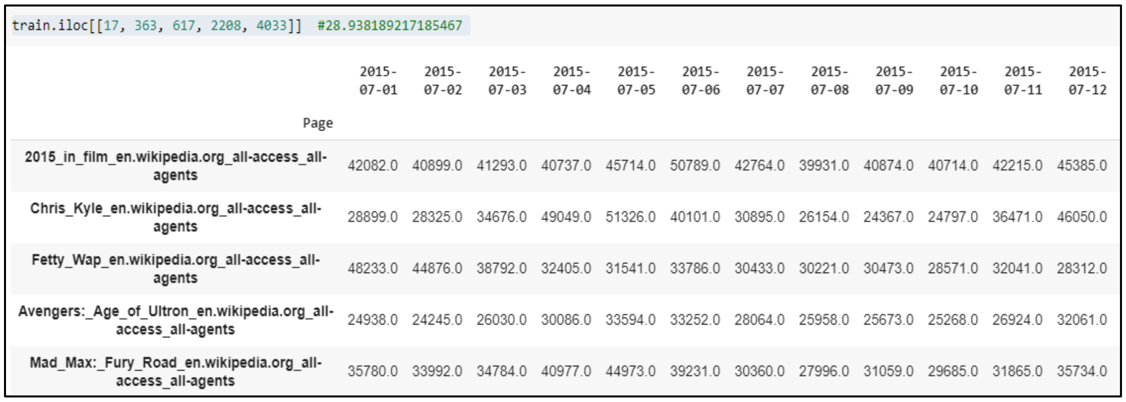


```
[11, 606, 2231, 3517, 3998]
19.948218980990948
[17, 363, 617, 2208, 4033]
28.938189217185467
[44, 45, 47, 1370, 2690]
22.418152946555306
[88, 1930, 2284, 3114, 3618]
18.382130681760092
[96, 732, 1396, 2586, 2970]
15.152038677179998
[113, 776, 848, 3003, 3225]
17.495222771040524
[122, 689, 2300, 3675, 3965]
15.394529675552867
```

Figure 19: Snapshot of the sum of the intra-cluster distance

The below results are sorted in the descending order of the sum of the Euclidean(sum-euc) distances. The higher the sum of Euclidean distance, the smaller the intracluster similarity. Thus, ideally, clusters with smaller Euclidean distances ought to fetch better results.





```
train.iloc[[17, 363, 617, 2208, 4033]]  #28.938189217185467
```

Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10	2015-07-11	2015-07-12
2015_in_film_en.wikipedia.org_all-access_all-agents	42082.0	40899.0	41293.0	40737.0	45714.0	50789.0	42764.0	39931.0	40874.0	40714.0	42215.0	45385.0
Chris_Kyle_en.wikipedia.org_all-access_all-agents	28899.0	28325.0	34676.0	49049.0	51326.0	40101.0	30895.0	26154.0	24367.0	24797.0	36471.0	46050.0
Fetty_Wap_en.wikipedia.org_all-access_all-agents	48233.0	44876.0	38792.0	32405.0	31541.0	33786.0	30433.0	30221.0	30473.0	28571.0	32041.0	28312.0
Avengers:_Age_of_Ultron_en.wikipedia.org_all-access_all-agents	24938.0	24245.0	26030.0	30086.0	33594.0	33252.0	28064.0	25958.0	25673.0	25268.0	26924.0	32061.0
Mad_Max:_Fury_Road_en.wikipedia.org_all-access_all-agents	35780.0	33992.0	34784.0	40977.0	44973.0	39231.0	30360.0	27996.0	31059.0	29685.0	31865.0	35734.0

Figure 20: Snapshot of a cluster with five elements - one

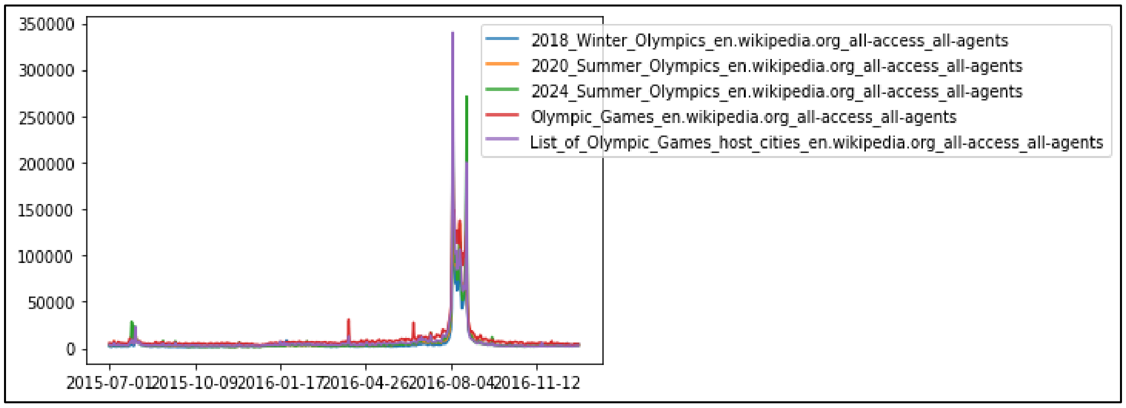
Observations:

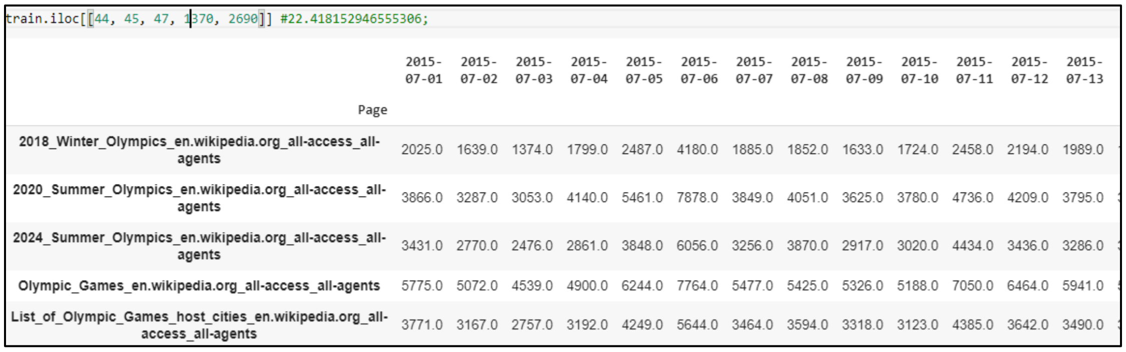
a. Cluster with the highest sum-euc distance

b. Mad Max and Avengers are released in the year 2015 which is linked to the webpage in the top row (all movies released in 2015)

c. A movie based on Chris Kyle is released in 2014

d. Fetty Wap is an anomaly where there is no apparent connection





```
train.iloc[[44, 45, 47, 1370, 2690]] #22.418152946555306;
```

Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10	2015-07-11	2015-07-12	2015-07-13
2018_Winter_Olympics_en.wikipedia.org_all-access_all-agents	2025.0	1639.0	1374.0	1799.0	2487.0	4180.0	1885.0	1852.0	1633.0	1724.0	2458.0	2194.0	1989.0
2020_Summer_Olympics_en.wikipedia.org_all-access_all-agents	3866.0	3287.0	3053.0	4140.0	5461.0	7878.0	3849.0	4051.0	3625.0	3780.0	4736.0	4209.0	3795.0
2024_Summer_Olympics_en.wikipedia.org_all-access_all-agents	3431.0	2770.0	2476.0	2861.0	3848.0	6056.0	3256.0	3870.0	2917.0	3020.0	4434.0	3436.0	3286.0
Olympic_Games_en.wikipedia.org_all-access_all-agents	5775.0	5072.0	4539.0	4900.0	6244.0	7764.0	5477.0	5425.0	5326.0	5188.0	7050.0	6464.0	5941.0
List_of_Olympic_Games_host_cities_en.wikipedia.org_all-access_all-agents	3771.0	3167.0	2757.0	3192.0	4249.0	5644.0	3464.0	3594.0	3318.0	3123.0	4385.0	3642.0	3490.0

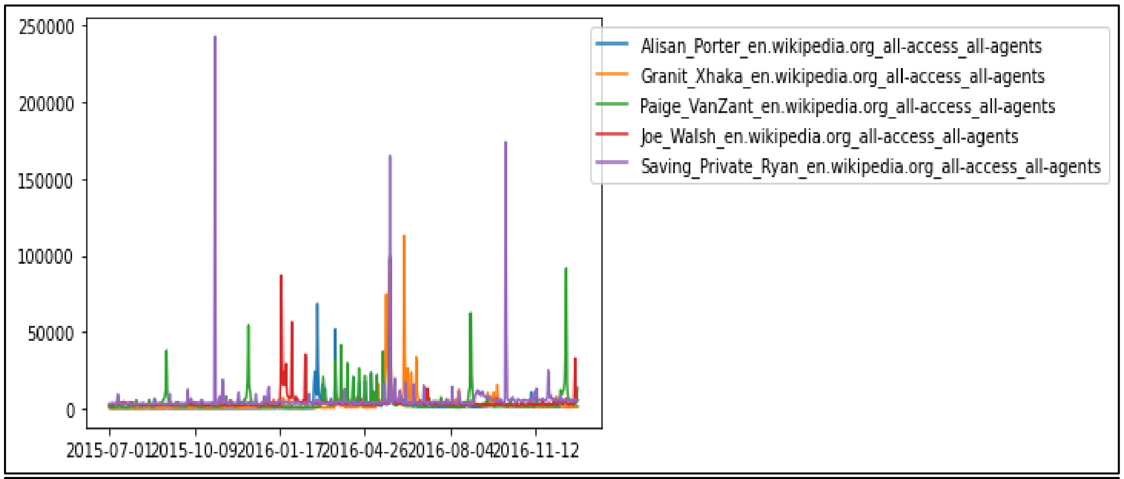
Figure 21: Snapshot of a cluster with five elements - two

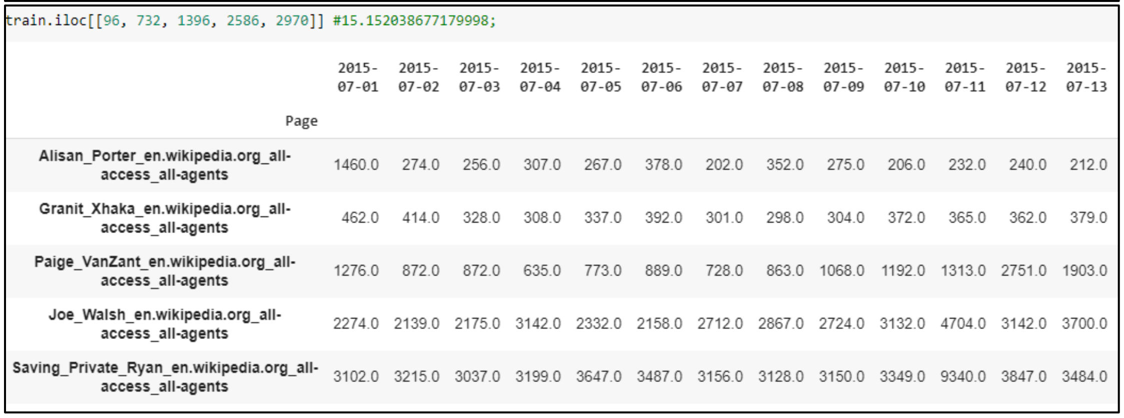
Observations:

a. Cluster with the second-highest sum-euc distance

b. There is an obvious common theme - the Olympics.

c. Cannot spot any anomaly with this result





```
train.iloc[[96, 732, 1396, 2586, 2970]] #15.152038677179998;
```

Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10	2015-07-11	2015-07-12	2015-07-13
Alisan_Porter_en.wikipedia.org_all-access_all-agents	1460.0	274.0	256.0	307.0	267.0	378.0	202.0	352.0	275.0	206.0	232.0	240.0	212.0
Granit_Xhaka_en.wikipedia.org_all-access_all-agents	462.0	414.0	328.0	308.0	337.0	392.0	301.0	298.0	304.0	372.0	365.0	362.0	379.0
Paige_VanZant_en.wikipedia.org_all-access_all-agents	1276.0	872.0	872.0	635.0	773.0	889.0	728.0	863.0	1068.0	1192.0	1313.0	2751.0	1903.0
Joe_Walsh_en.wikipedia.org_all-access_all-agents	2274.0	2139.0	2175.0	3142.0	2332.0	2158.0	2712.0	2867.0	2724.0	3132.0	4704.0	3142.0	3700.0
Saving_Private_Ryan_en.wikipedia.org_all-access_all-agents	3102.0	3215.0	3037.0	3199.0	3647.0	3487.0	3156.0	3128.0	3150.0	3349.0	9340.0	3847.0	3484.0

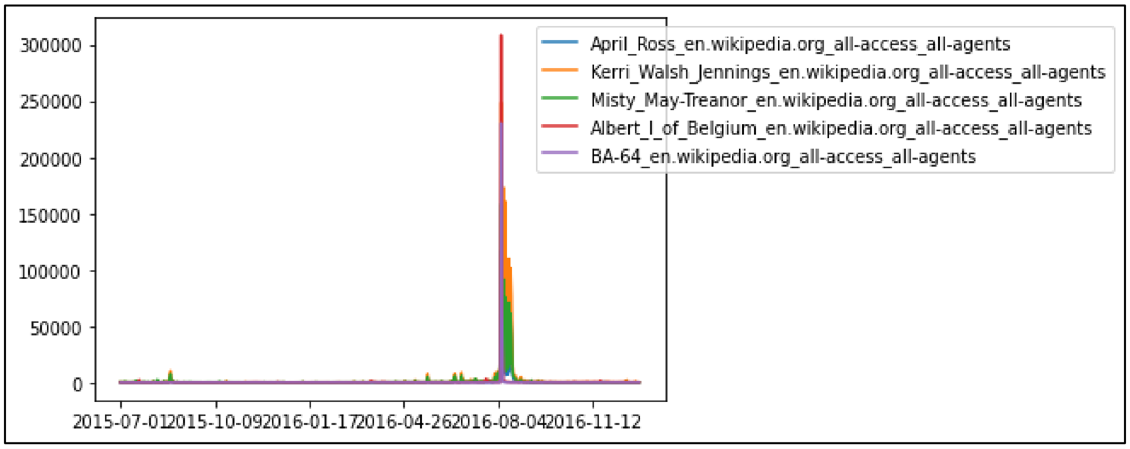
Figure 22: Snapshot of a cluster with five elements - three

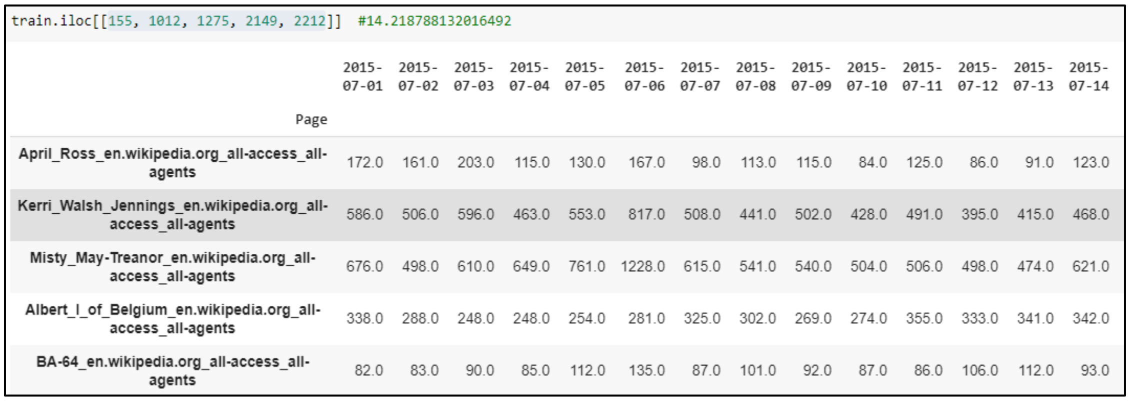
Observations:

a. Cluster with the second-lowest sum-euc distance. Even though the cluster has a high similarity score, its apparent performance is worse than the above cluster.

b. Alison Porter and Joe Walsh are musicians, the last record is a movie, and whereas others are sports players. No common theme.

c. Even though there is no common theme, it appears that the general interests on these Wikipedia pages pique around the same time.





```
train.iloc[[155, 1012, 1275, 2149, 2212]]  #14.218788132016492
```

Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10	2015-07-11	2015-07-12	2015-07-13	2015-07-14
April_Ross_en.wikipedia.org_all-access_all-agents	172.0	161.0	203.0	115.0	130.0	167.0	98.0	113.0	115.0	84.0	125.0	86.0	91.0	123.0
Kerri_Walsh_Jennings_en.wikipedia.org_all-access_all-agents	586.0	506.0	596.0	463.0	553.0	817.0	508.0	441.0	502.0	428.0	491.0	395.0	415.0	468.0
Misty_May-Treanor_en.wikipedia.org_all-access_all-agents	676.0	498.0	610.0	649.0	761.0	1228.0	615.0	541.0	540.0	504.0	506.0	498.0	474.0	621.0
Albert_I_of_Belgium_en.wikipedia.org_all-access_all-agents	338.0	288.0	248.0	248.0	254.0	281.0	325.0	302.0	269.0	274.0	355.0	333.0	341.0	342.0
BA-64_en.wikipedia.org_all-access_all-agents	82.0	83.0	90.0	85.0	112.0	135.0	87.0	101.0	92.0	87.0	86.0	106.0	112.0	93.0

Figure 23: Snapshot of a cluster with five elements - four

Observations:

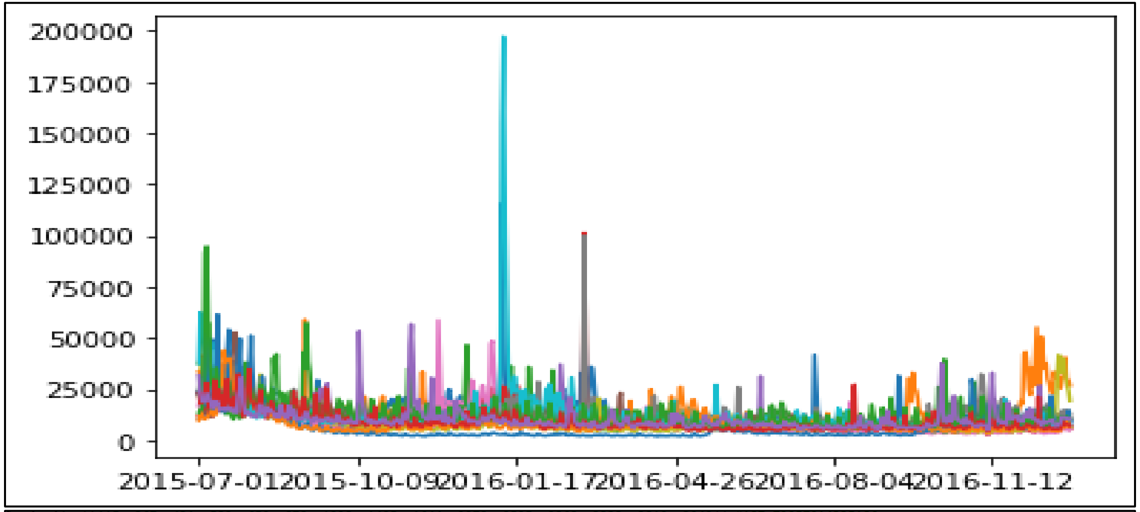
a. Cluster with the lowest sum-euc distance

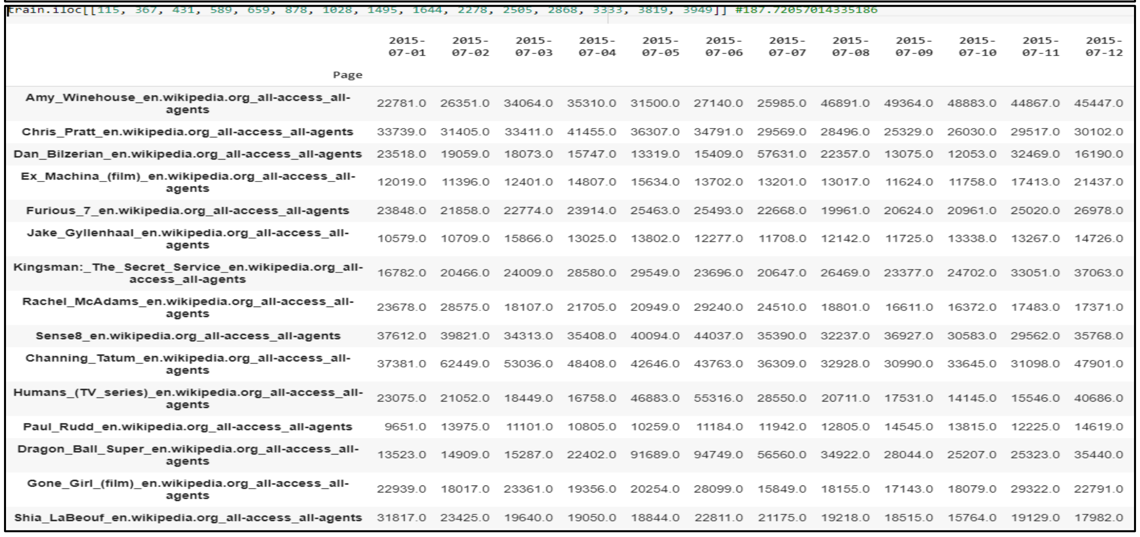
b. Similarity: April Ross, Kerri Walsh, and Misty May have a personal relationship in some fashion. Albert I (of Belgium) is a figure from World War 1 where are BA 64 is a Soviet armored car from world war II.

c. Anomaly: Two different clusters exist in the same cluster. The top 3 rows are people related to each other while the bottom two rows are about World War.

1.6 The cluster of other than five elements

15 Elements





```
train.iloc[[115, 367, 431, 589, 659, 878, 1028, 1495, 1644, 2278, 2505, 2868, 3333, 3819, 3949]] #187.72057014335186
```

Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10	2015-07-11	2015-07-12
Amy_Winehouse_en.wikipedia.org_all-access_all-agents	22781.0	26351.0	34064.0	35310.0	31500.0	27140.0	25985.0	46891.0	49364.0	48883.0	44867.0	45447.0
Chris_Pratt_en.wikipedia.org_all-access_all-agents	33739.0	31405.0	33411.0	41455.0	36307.0	34791.0	29569.0	28496.0	25329.0	26030.0	29517.0	30102.0
Dan_Bilzerian_en.wikipedia.org_all-access_all-agents	23518.0	19059.0	18073.0	15747.0	13319.0	15409.0	57631.0	22357.0	13075.0	12053.0	32469.0	16190.0
Ex_Machina_(film)_en.wikipedia.org_all-access_all-agents	12019.0	11396.0	12401.0	14807.0	15634.0	13702.0	13201.0	13017.0	11624.0	11758.0	17413.0	21437.0
Furious_7_en.wikipedia.org_all-access_all-agents	23848.0	21858.0	22774.0	23914.0	25463.0	25493.0	22668.0	19961.0	20624.0	20961.0	25020.0	26978.0
Jake_Gyllenhaal_en.wikipedia.org_all-access_all-agents	10579.0	10709.0	15866.0	13025.0	13802.0	12277.0	11708.0	12142.0	11725.0	13338.0	13267.0	14726.0
Kingsman:_The_Secret_Service_en.wikipedia.org_all-access_all-agents	16782.0	20466.0	24009.0	28580.0	29549.0	23696.0	20647.0	26469.0	23377.0	24702.0	33051.0	37063.0
Rachel_McAdams_en.wikipedia.org_all-access_all-agents	23678.0	28575.0	18107.0	21705.0	20949.0	29240.0	24510.0	18801.0	16611.0	16372.0	17483.0	17371.0
Sense8_en.wikipedia.org_all-access_all-agents	37612.0	39821.0	34313.0	35408.0	40094.0	44037.0	35390.0	32237.0	36927.0	30583.0	29562.0	35768.0
Channing_Tatum_en.wikipedia.org_all-access_all-agents	37381.0	62449.0	53036.0	48408.0	42646.0	43763.0	36309.0	32928.0	30990.0	33645.0	31098.0	47901.0
Humans_(TV_series)_en.wikipedia.org_all-access_all-agents	23075.0	21052.0	18449.0	16758.0	46883.0	55316.0	28550.0	20711.0	17531.0	14145.0	15546.0	40686.0
Paul_Rudd_en.wikipedia.org_all-access_all-agents	9651.0	13975.0	11101.0	10805.0	10259.0	11184.0	11942.0	12805.0	14545.0	13815.0	12225.0	14619.0
Dragon_Ball_Super_en.wikipedia.org_all-access_all-agents	13523.0	14909.0	15287.0	22402.0	91689.0	94749.0	56560.0	34922.0	28044.0	25207.0	25323.0	35440.0
Gone_Girl_(film)_en.wikipedia.org_all-access_all-agents	22939.0	18017.0	23361.0	19356.0	20254.0	28099.0	15849.0	18155.0	17143.0	18079.0	29322.0	22791.0
Shia_LaBeouf_en.wikipedia.org_all-access_all-agents	31817.0	23425.0	19640.0	19050.0	18844.0	22811.0	21175.0	19218.0	18515.0	15764.0	19129.0	17982.0

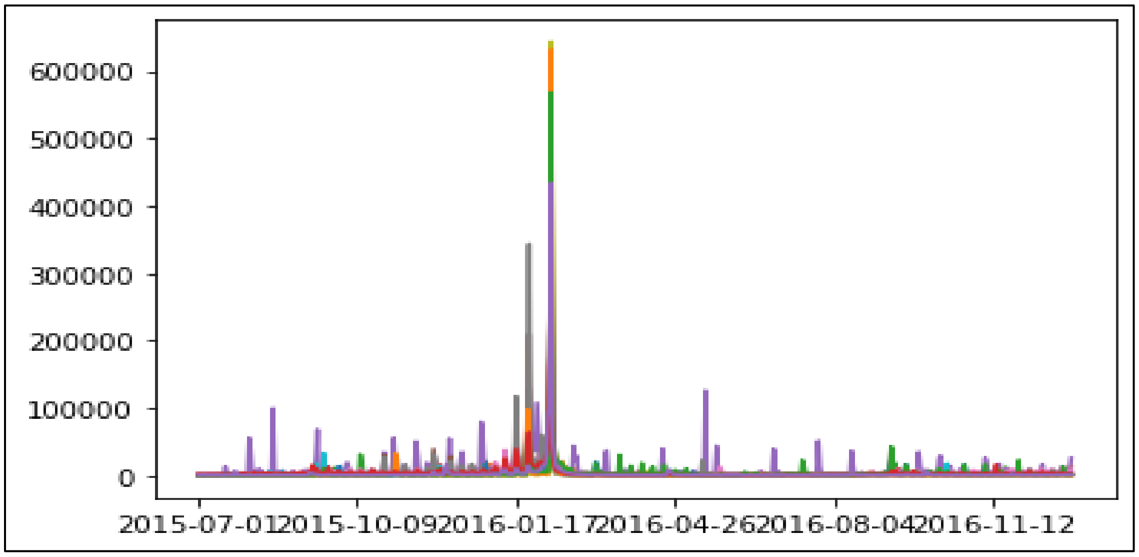
Figure 24: Snapshot of a cluster with fifteen elements - one

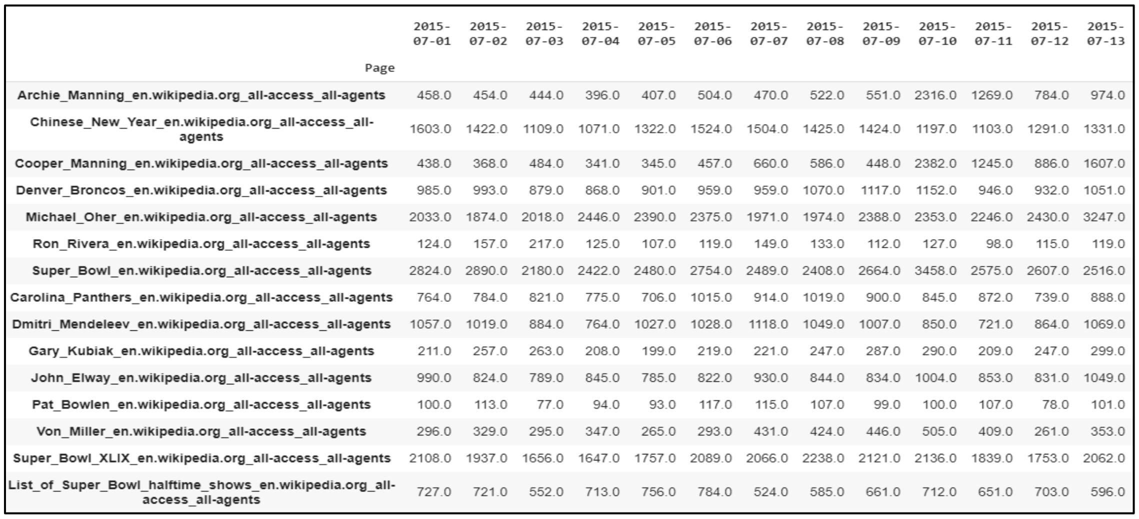
Observations

a. Cluster with the highest sum-euc distance, among 15 elements cluster

b. All the records have a typical movie or a TV show theme

c. During the holiday season (Dec-Jan), there is a huge spike followed by another spike during the spring break season. Indicating how the popularity of movies surge during that time.





Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10	2015-07-11	2015-07-12	2015-07-13
Archie_Manning_en.wikipedia.org_all-access_all-agents	458.0	454.0	444.0	396.0	407.0	504.0	470.0	522.0	551.0	2316.0	1269.0	784.0	974.0
Chinese_New_Year_en.wikipedia.org_all-access_all-agents	1603.0	1422.0	1109.0	1071.0	1322.0	1524.0	1504.0	1425.0	1424.0	1197.0	1103.0	1291.0	1331.0
Cooper_Manning_en.wikipedia.org_all-access_all-agents	438.0	368.0	484.0	341.0	345.0	457.0	660.0	586.0	448.0	2382.0	1245.0	886.0	1607.0
Denver_Broncos_en.wikipedia.org_all-access_all-agents	985.0	993.0	879.0	868.0	901.0	959.0	959.0	1070.0	1117.0	1152.0	946.0	932.0	1051.0
Michael_Oher_en.wikipedia.org_all-access_all-agents	2033.0	1874.0	2018.0	2446.0	2390.0	2375.0	1971.0	1974.0	2388.0	2353.0	2246.0	2430.0	3247.0
Ron_Rivera_en.wikipedia.org_all-access_all-agents	124.0	157.0	217.0	125.0	107.0	119.0	149.0	133.0	112.0	127.0	98.0	115.0	119.0
Super_Bowl_en.wikipedia.org_all-access_all-agents	2824.0	2890.0	2180.0	2422.0	2480.0	2754.0	2489.0	2408.0	2664.0	3458.0	2575.0	2607.0	2516.0
Carolina_Panthers_en.wikipedia.org_all-access_all-agents	764.0	784.0	821.0	775.0	706.0	1015.0	914.0	1019.0	900.0	845.0	872.0	739.0	888.0
Dmitri_Mendeleev_en.wikipedia.org_all-access_all-agents	1057.0	1019.0	884.0	764.0	1027.0	1028.0	1118.0	1049.0	1007.0	850.0	721.0	864.0	1069.0
Gary_Kubiak_en.wikipedia.org_all-access_all-agents	211.0	257.0	263.0	208.0	199.0	219.0	221.0	247.0	287.0	290.0	209.0	247.0	299.0
John_Elway_en.wikipedia.org_all-access_all-agents	990.0	824.0	789.0	845.0	785.0	822.0	930.0	844.0	834.0	1004.0	853.0	831.0	1049.0
Pat_Bowlen_en.wikipedia.org_all-access_all-agents	100.0	113.0	77.0	94.0	93.0	117.0	115.0	107.0	99.0	100.0	107.0	78.0	101.0
Von_Miller_en.wikipedia.org_all-access_all-agents	296.0	329.0	295.0	347.0	265.0	293.0	431.0	424.0	446.0	505.0	409.0	261.0	353.0
Super_Bowl_XLIX_en.wikipedia.org_all-access_all-agents	2108.0	1937.0	1656.0	1647.0	1757.0	2089.0	2066.0	2238.0	2121.0	2136.0	1839.0	1753.0	2062.0
List_of_Super_Bowl_halftime_shows_en.wikipedia.org_all-access_all-agents	727.0	721.0	552.0	713.0	756.0	784.0	524.0	585.0	661.0	712.0	651.0	703.0	596.0

Figure 25: Snapshot of a cluster with fifteen elements - two

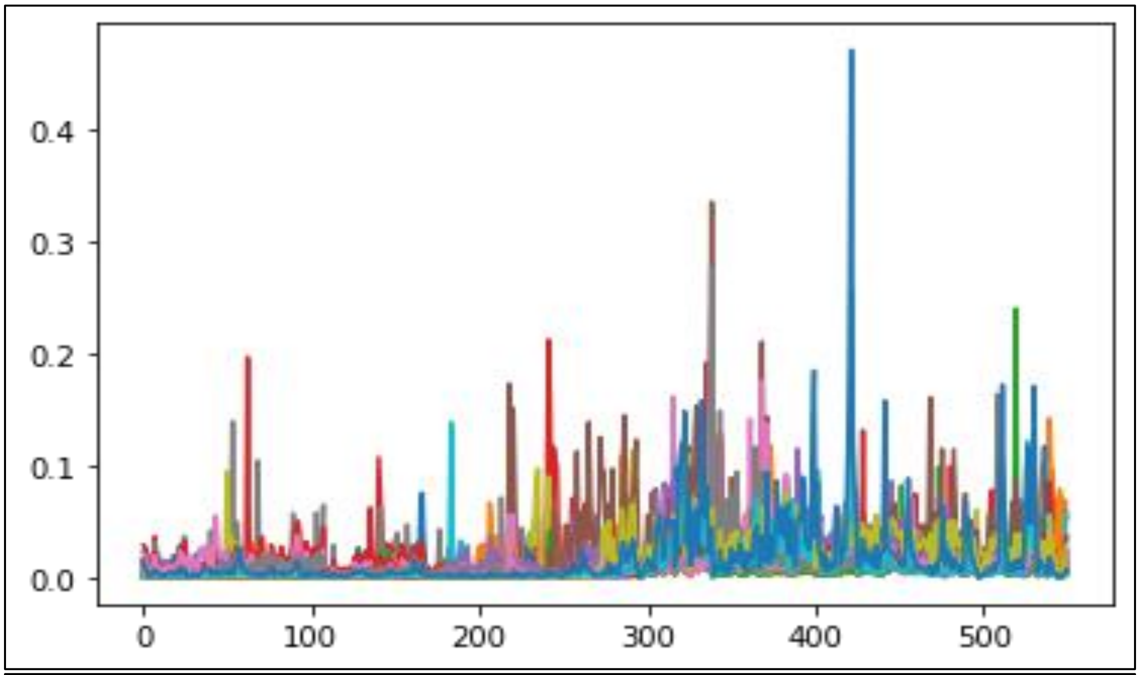
Observations

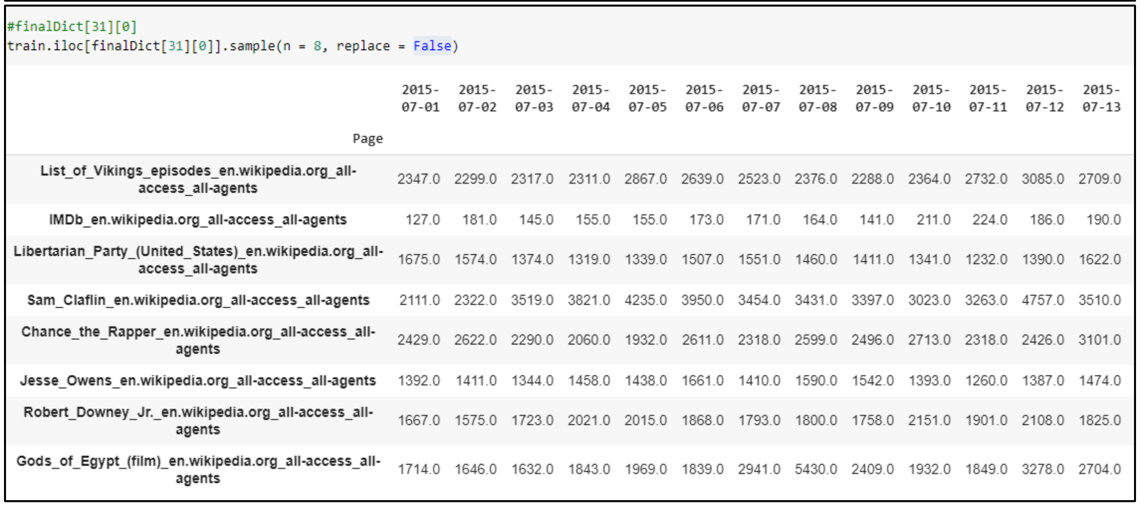
a. Cluster with the lowest sum-euc distance, among 15 elements cluster

b. All the records have a common sports theme, specifically American Football

c. During the holiday season (Dec-Jan), there is a huge spike followed by another spike during the summer. Indicating how the popularity of sports surge at that time

31 elements





```
#finalDict[31][0]
train.iloc[finalDict[31][0]].sample(n = 8, replace = False)
```

Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10	2015-07-11	2015-07-12	2015-07-13
List_of_Vikings_episodes_en.wikipedia.org_all-access_all-agents	2347.0	2299.0	2317.0	2311.0	2867.0	2639.0	2523.0	2376.0	2288.0	2364.0	2732.0	3085.0	2709.0
IMDb_en.wikipedia.org_all-access_all-agents	127.0	181.0	145.0	155.0	155.0	173.0	171.0	164.0	141.0	211.0	224.0	186.0	190.0
Libertarian_Party_(United_States)_en.wikipedia.org_all-access_all-agents	1675.0	1574.0	1374.0	1319.0	1339.0	1507.0	1551.0	1460.0	1411.0	1341.0	1232.0	1390.0	1622.0
Sam_Claflin_en.wikipedia.org_all-access_all-agents	2111.0	2322.0	3519.0	3821.0	4235.0	3950.0	3454.0	3431.0	3397.0	3023.0	3263.0	4757.0	3510.0
Chance_the_Rapper_en.wikipedia.org_all-access_all-agents	2429.0	2622.0	2290.0	2060.0	1932.0	2611.0	2318.0	2599.0	2496.0	2713.0	2318.0	2426.0	3101.0
Jesse_Owens_en.wikipedia.org_all-access_all-agents	1392.0	1411.0	1344.0	1458.0	1438.0	1661.0	1410.0	1590.0	1542.0	1393.0	1260.0	1387.0	1474.0
Robert_Downey_Jr._en.wikipedia.org_all-access_all-agents	1667.0	1575.0	1723.0	2021.0	2015.0	1868.0	1793.0	1800.0	1758.0	2151.0	1901.0	2108.0	1825.0
Gods_of_Egypt_(film)_en.wikipedia.org_all-access_all-agents	1714.0	1646.0	1632.0	1843.0	1969.0	1839.0	2941.0	5430.0	2409.0	1932.0	1849.0	3278.0	2704.0

Figure 26: Snapshot of a cluster with 31 elements

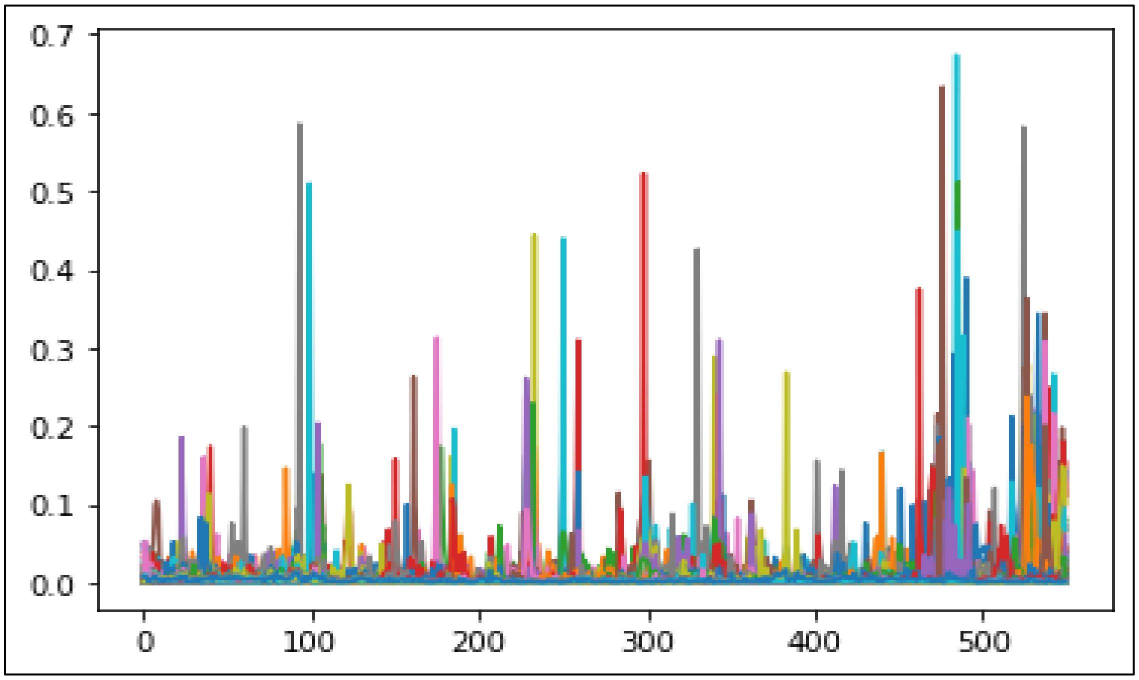
Observations:

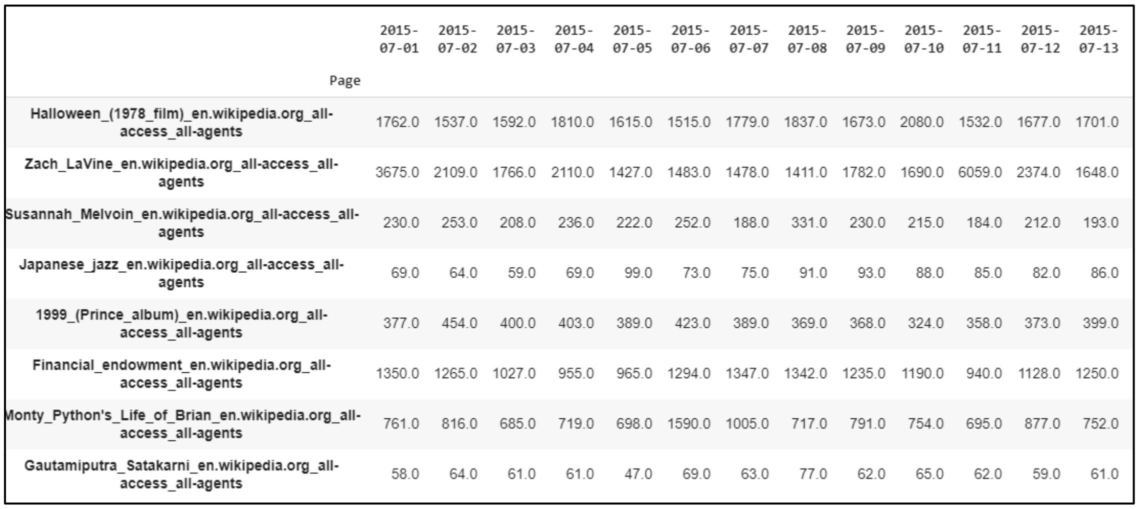
a. Randomly selected eight elements out of 31 elements

b. The topics are movies, sports, and politics.

c. The frequency and count of page visits increase after the summer of 2016, indicating that the popularity of these topics surged after a certain point in time.

281 elements





Page	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10	2015-07-11	2015-07-12	2015-07-13
Halloween_(1978_film)_en.wikipedia.org_all-access_all-agents	1762.0	1537.0	1592.0	1810.0	1615.0	1515.0	1779.0	1837.0	1673.0	2080.0	1532.0	1677.0	1701.0
Zach_LaVine_en.wikipedia.org_all-access_all-agents	3675.0	2109.0	1766.0	2110.0	1427.0	1483.0	1478.0	1411.0	1782.0	1690.0	6059.0	2374.0	1648.0
Susannah_Melvoin_en.wikipedia.org_all-access_all-agents	230.0	253.0	208.0	236.0	222.0	252.0	188.0	331.0	230.0	215.0	184.0	212.0	193.0
Japanese_jazz_en.wikipedia.org_all-access_all-agents	69.0	64.0	59.0	69.0	99.0	73.0	75.0	91.0	93.0	88.0	85.0	82.0	86.0
1999_(Prince_album)_en.wikipedia.org_all-access_all-agents	377.0	454.0	400.0	403.0	389.0	423.0	389.0	369.0	368.0	324.0	358.0	373.0	399.0
Financial_endowment_en.wikipedia.org_all-access_all-agents	1350.0	1265.0	1027.0	955.0	965.0	1294.0	1347.0	1342.0	1235.0	1190.0	940.0	1128.0	1250.0
Monty_Python's_Life_of_Brian_en.wikipedia.org_all-access_all-agents	761.0	816.0	685.0	719.0	698.0	1590.0	1005.0	717.0	791.0	754.0	695.0	877.0	752.0
Gautamiputra_Satakarni_en.wikipedia.org_all-access_all-agents	58.0	64.0	61.0	61.0	47.0	69.0	63.0	77.0	62.0	65.0	62.0	59.0	61.0

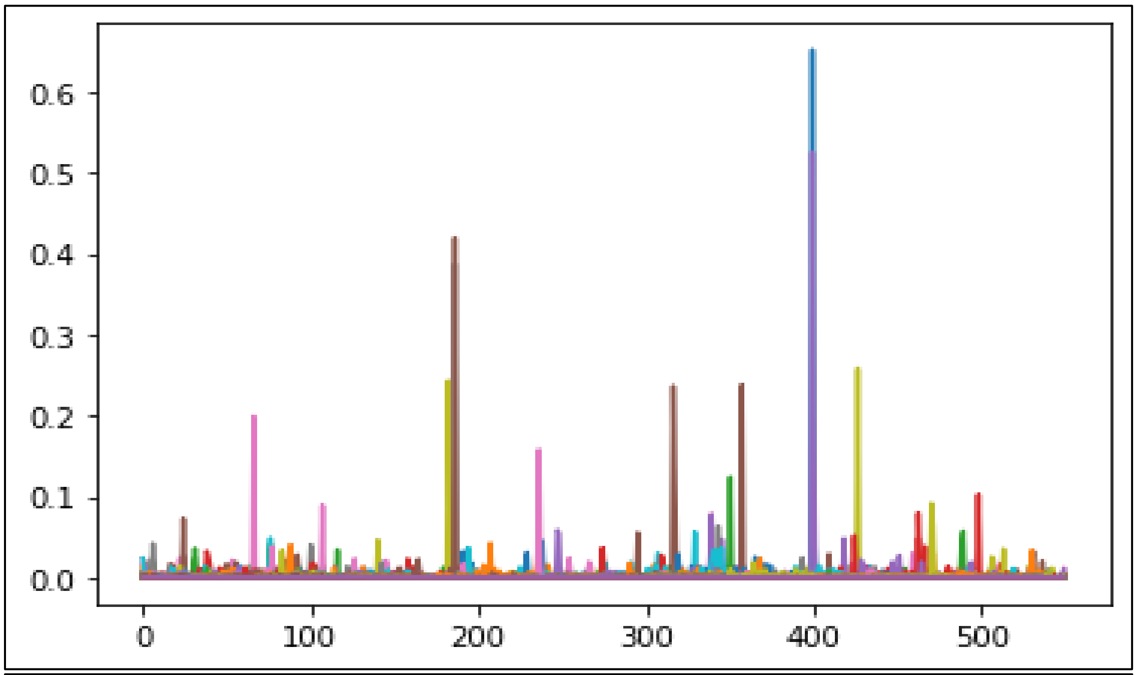
Figure 27: Snapshot of a cluster with 281 elements

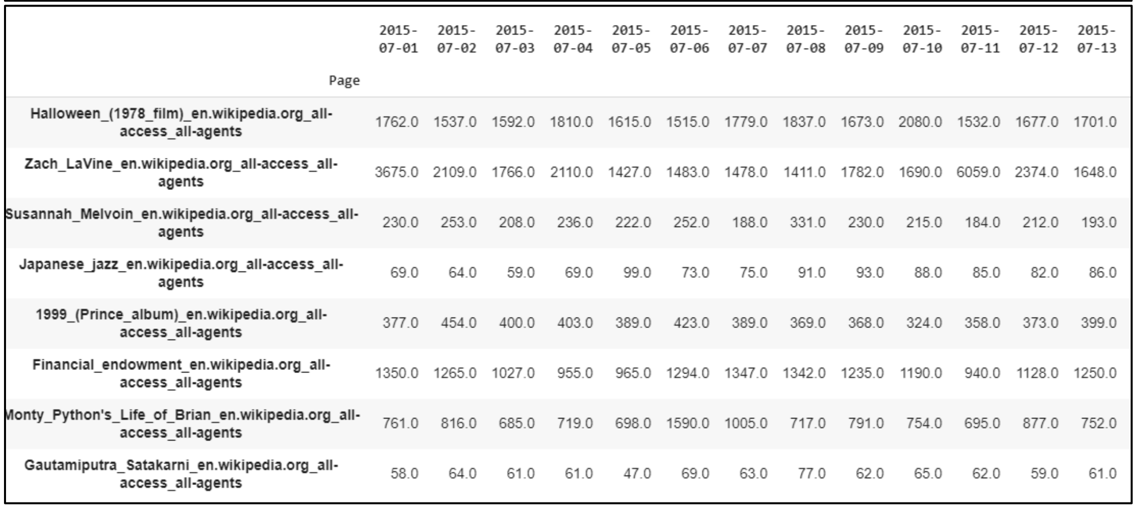
Observations:

a. Randomly selected eight elements out of 281 elements

b. Even though there is no common theme, it appears that the general interests on these Wikipedia pages pique around the same time. The rise and fall in general interest are consistent throughout the entire time period.

1555 elements





	2015-07-01	2015-07-02	2015-07-03	2015-07-04	2015-07-05	2015-07-06	2015-07-07	2015-07-08	2015-07-09	2015-07-10	2015-07-11	2015-07-12	2015-07-13
Page													
Halloween_(1978_film)_en.wikipedia.org_all-access_all-agents	1762.0	1537.0	1592.0	1810.0	1615.0	1515.0	1779.0	1837.0	1673.0	2080.0	1532.0	1677.0	1701.0
Zach_LaVine_en.wikipedia.org_all-access_all-agents	3675.0	2109.0	1766.0	2110.0	1427.0	1483.0	1478.0	1411.0	1782.0	1690.0	6059.0	2374.0	1648.0
Susannah_Melvoin_en.wikipedia.org_all-access_all-agents	230.0	253.0	208.0	236.0	222.0	252.0	188.0	331.0	230.0	215.0	184.0	212.0	193.0
Japanese_jazz_en.wikipedia.org_all-access_all-agents	69.0	64.0	59.0	69.0	99.0	73.0	75.0	91.0	93.0	88.0	85.0	82.0	86.0
1999_(Prince_album)_en.wikipedia.org_all-access_all-agents	377.0	454.0	400.0	403.0	389.0	423.0	389.0	369.0	368.0	324.0	358.0	373.0	399.0
Financial_endowment_en.wikipedia.org_all-access_all-agents	1350.0	1265.0	1027.0	955.0	965.0	1294.0	1347.0	1342.0	1235.0	1190.0	940.0	1128.0	1250.0
Monty_Python's_Life_of_Brian_en.wikipedia.org_all-access_all-agents	761.0	816.0	685.0	719.0	698.0	1590.0	1005.0	717.0	791.0	754.0	695.0	877.0	752.0
Gautamiputra_Satakarni_en.wikipedia.org_all-access_all-agents	58.0	64.0	61.0	61.0	47.0	69.0	63.0	77.0	62.0	65.0	62.0	59.0	61.0

Figure 28: Snapshot of a cluster with the highest number of elements

Observations:

a. Randomly selected eight elements out of 1555 elements

b. Relatively there is a lesser number of associated peaks than the cluster of 281 elements. Additionally, there is no apparent common theme.

c. Suggesting that if more clusters were requested, these pages would not have been grouped in one cluster.

DISCUSSION

The results indicate that pages in each cluster have an intrinsic relation between them. They further show at what seasons or months the relationship is at its peak. Such a piece of information on association with another time series can help improve any univariate time series model expand to a multivariate time series model. Forecasting of one series can be improved by adding data (whether lagged or current) from the associated/linked series.

As indicated in the results, there are two types of clusters formed. One in which the topics are entirely related to each other, and there is a common theme among the clusters and the other one, where some wiki pages receive seasonal (summer or during holidays) interest. Additionally, there will be some topics that are trending only for one year, i.e., it is popular in the year 2016, but the popularity fades away in consecutive years.

Coming to the number of elements in a cluster, the cluster of fifteen performed better on grouping similar topics together than the cluster of five, which is slightly unexpected. Moreover, even in a cluster of five, the ones with higher intra-cluster distance (less similarity) grouped topics better than the ones with a higher similarity score. It suggests that there is a sweet spot in deciding the number of clusters and the number of elements in each cluster.

The most prevalent themes appear to be sports, movies (and TV shows), including the cast, and politics. This is expected since these are widely prevalent subjects in almost all geographies. But a Wikipedia page could be of relevance (or importance) in one geography but not in others. Thus making it more complicated where there is no geographical information.

Small clusters are more discernible than large clusters. But, the large clusters could carry some information that is not apparent. The data about other elements in a cluster can be fed into one element for better forecasting. In order to understand how elements in large clusters influence each other, the data from all the other elements can be added as an additional variable. It will make it a multivariate time series model, albeit a hefty one.

As said earlier, the primary motivation is exploring the time-series models that use the non-traditional methods of forecasting. In that regard, the below models are explored that could be enhanced with the cluster data obtained above.

1.7 Sequence to Sequence model:

Below is a proof of concept (POC) that is picked as-is from one of the entries in the Kaggle competition. (Eddy, 2018). A repeat experiment was performed but using the project's modified dataset. The below figure gives a brief idea of what entails a sequence to sequence (seq2seq) model using a recurring neural network (RNN) layer. The RNN layer uses Long short term memory (LSTM) to store an internal state that remembers the past information and uses it to predict the future.

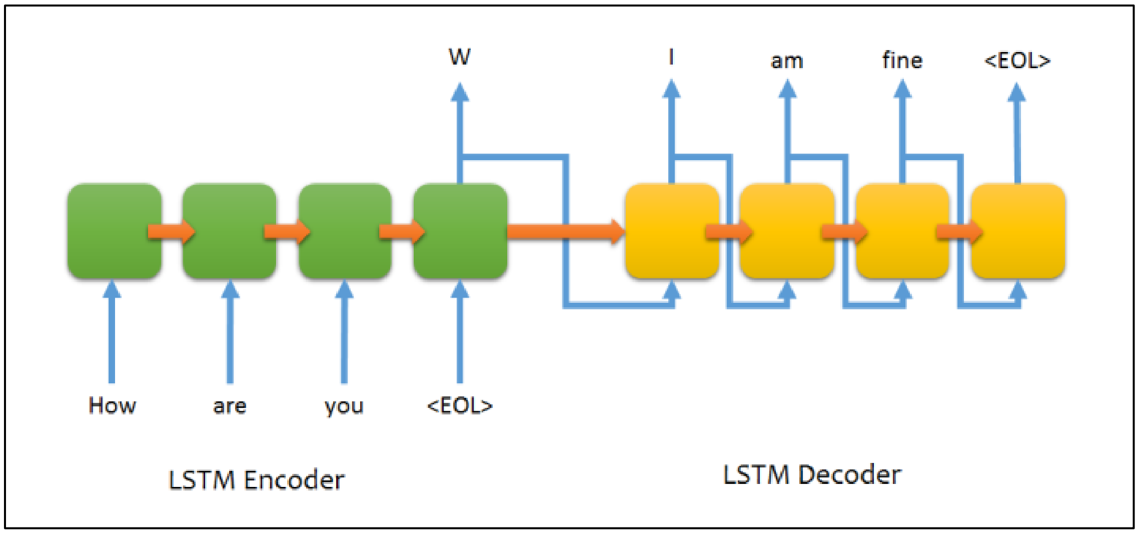


Figure 29: A seq2seq model with LSTM layer (Eddy, 2018)

In this POC, it is proposed to do a multi-step prediction., i.e., predict the next 14 days of Wikipedia page visits instead of just one day. The below figure (Suilin, 2018) suggests how to split the training and validation set.

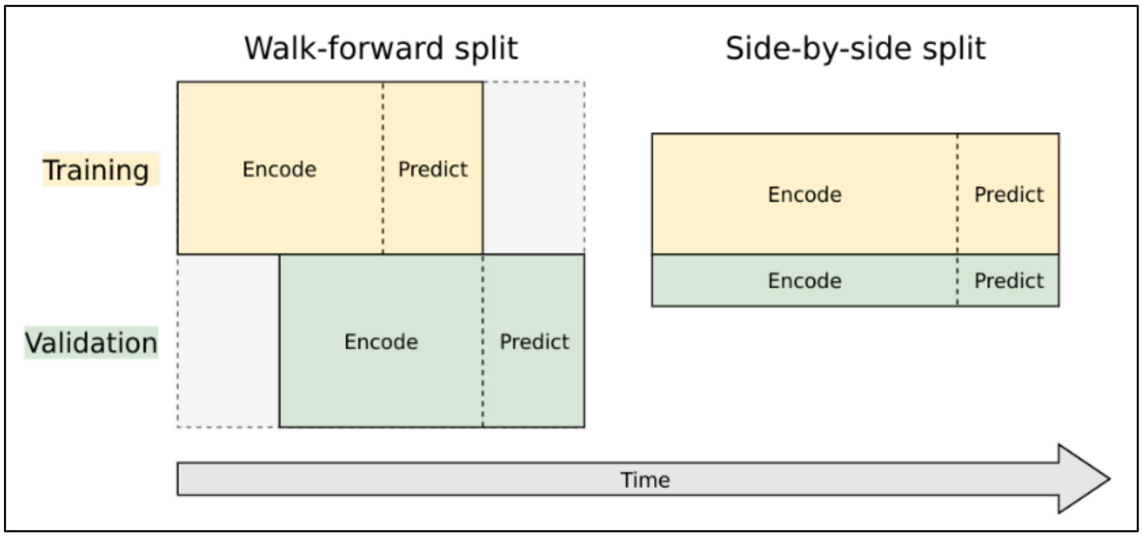
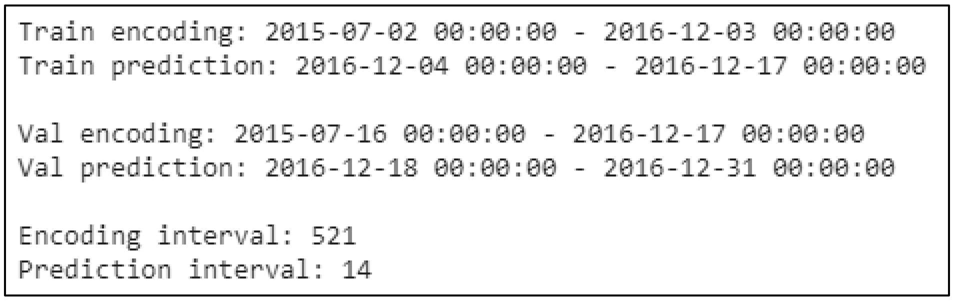


Figure 30: Train-Test split of a multi-step time series prediction (Suilin, 2018)

It suggests training the encoder part of the seq2seq model with the train-test split, as shown in the below figure.



```
Train encoding: 2015-07-02 00:00:00 - 2016-12-03 00:00:00
Train prediction: 2016-12-04 00:00:00 - 2016-12-17 00:00:00

Val encoding: 2015-07-16 00:00:00 - 2016-12-17 00:00:00
Val prediction: 2016-12-18 00:00:00 - 2016-12-31 00:00:00

Encoding interval: 521
Prediction interval: 14
```

Figure 31: Snapshot of the data's train-test split windows

Once the encoder model is configured, it is trained using my dataset.

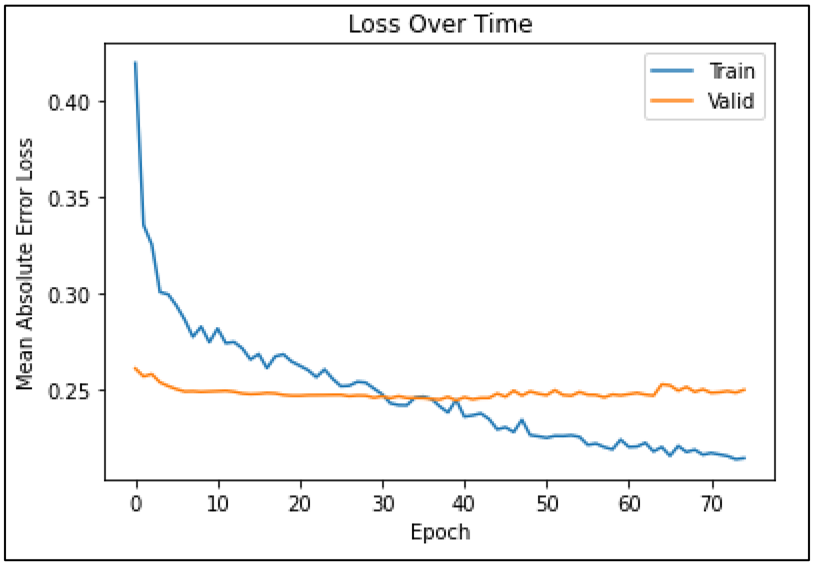


Figure 32: Plot of the epoch runs of the encoder model

After this comes the decoder step. The output from the encoder step (the state of the encoder) is fed into the decoder model. Such feed accounts and captures historical information about a time series, and it uses that historical information to make future predictions. As suggested in Figure 29, the decoder is used to make predictions one step at a time. It means that the output of step one in the decoder is sent as input to predict step two in the decoder sequence.

The above POC can be expanded to incorporate additional time-series data. Instead of only sending the decoder or encoder output to the decoder sequence, additional inputs based on the hierarchical flat clustering similarity could be fed. The inputs could be live time-series data from a few other selected time series.

1.8 Adding a regressor to the univariate time series model:

Use automated open source libraries like facebook's (FB) Prophet. The FB Prophet is an easy to use python library, that does not require knowledge of time series analysis. Just feed the series into the library, and it automatically decomposes it into its basic elements. It is just a one-line command call that makes it very simple.

To such a model, it is again effortlessly possible to add additional regressor variables (which could be data values from other time series) so that it becomes a multivariate model. These regressor variables can be chosen based on output from hierarchical clustering. Once a model for time series data is created, a function could be created to call the model recursively to make multi-step forecasting.

Finally, the same function could be called recursively for all the time series in the dataset. FB prophet is only suggested as one of the libraries since its currently popular, any other licensed libraries could also be used in the manner suggested as above.

LIMITATIONS AND FUTURE WORK

During the data preparation, it is seen that the dataset contains pages accessed by both the spiderbot and human interaction, and differentiating between them is difficult. It raises a possibility that a "drill-down" sort of crawl could have been performed. Thus, indicating an apparent connection between webpages which in reality are not of interest to human readers.

The dataset was abundant with themes such as movies, TV shows, sports, and politics. This could have influenced the model performance and will be interesting to see how adding a new unrelated theme with equal frequency could impact the model.

There are seasonal components, sudden trending wiki pages (and their subsequent demise), and similar topics or themes that one must understand in order to forecast accurately. Future work could involve incorporating the above Seq2Seq-LSTM model (with hierarchical clustering) but with a much larger dataset. The dataset should span across multiple years and be a continuous feed to improve the model continually. The model should be capable of identifying each of the three qualities, as mentioned above (sudden interest, seasonality, and topics).

The dataset used in this project is clean without any missing values or inconsistency, which is hardly the case. But, the model needs to be robust to handle missing data. Missing data can be filled with interpolation or other standard techniques. The real challenge will lie in creating an efficient pre-processing tool that can pick and choose the technique automatically, if not by manual selection.

A complete automated solution (or library in python) that pre-processes the data, performs clustering (thus multivariate), and finally forecasts data using seq2seq modeling (thus multi-step) can be beneficial. But the time and space complexity of such an automated solution increases exponentially with the increase in the number of time series, thus making it unfeasible (currently) for large and multiple time-series data.

CONCLUSION

This study is aimed at inferring the relationship between Wikipedia pages based on the number of visitations. It further explores the machine learning technique of hierarchical clustering to comprehend how pages could be grouped based on a similarity score. The results suggested that there is an optimal number of clusters (not by the lowest number of elements) that groups Wikipedia under a common theme. Currently, the results are analyzed manually. If a metric could be developed that analyzes the results automatically, it can help create a pipeline containing two models where the first model is based on clustering. The output could be fed into a second model (an example being a seq2seq with LSTM layer) to make a single or multi-step forecasting prediction.

BIBLIOGRAPHY

DTAI Research Group. (n.d.). *Time Series Distances*. Retrieved from zenodo.org: http://doi.org/10.5281/zenodo.3276100

Alam, M. (n.d.). *Multivariate time series forecasting*. Retrieved from towardsdatascience.com: https://towardsdatascience.com/multivariate-time-series-forecasting-653372b3db36

Bajak, A. (n.d.). *How to use hierarchical cluster analysis on time series data*. Retrieved from storybench.org: https://www.storybench.org/how-to-use-hierarchical-cluster-analysis-on-time-series-data/

Battula, M. (2019, Feb 20). *Time series forecasting with deep stacked unidirectional and bidirectional LSTMs*. Retrieved from towardsdatascience.com: https://towardsdatascience.com/time-series-forecasting-with-deep-stacked-unidirectional-and-bidirectional-lstms-de7c099bd918

Betts, R., Jones, C., Knight, J., Keeling, R., Kennedy, J., Wiltshire, A., . . . Aragao, L. (2018). A successful prediction of the record CO 2 rise associated with the 2015/2016 El Niño. *Philosophical Transactions of the Royal Society B: Biological Sciences*, p. 373(1760):20170301. doi:10.1098/rstb.2017.0301.

Brownlee, J. (2017, March 8). *4 Strategies for Multi-Step Time Series Forecasting*. Retrieved from machinelearningmastery.com: https://machinelearningmastery.com/multi-step-time-series-forecasting/

Brownlee, J. (2019, August 14). *A Gentle Introduction to LSTM Autoencoders*. Retrieved from machinelearningmastery.com: https://machinelearningmastery.com/lstm-autoencoders/

Brownlee, J. (2019, 08 21). *What Is Time Series Forecasting?* Retrieved from machinelearningmastery.com: https://machinelearningmastery.com/time-series-forecasting/

Brownlee, J. (2020, January 8). *How to Develop an Encoder-Decoder Model for Sequence-to-Sequence Prediction in Keras*. Retrieved from machinelearningmastery.com: https://machinelearningmastery.com/develop-encoder-decoder-model-sequence-sequence-prediction-keras/

Cassisi, C. M. (2012). Similarity measures and dimensionality reduction techniques for time series data mining. *Advances in data mining knowledge discovery and applications*, 71-96.

Cheong, J. H. (2019, May 13). *Four ways to quantify synchrony between time series data*. Retrieved from towardsdatascience.com: https://towardsdatascience.com/four-ways-to-quantify-synchrony-between-time-series-data-b99136c4a9c9

CO_2 and Greenhouse Gas Emissions. (n.d.). Retrieved from ourworldindata.org: https://ourworldindata.org/co2-and-other-greenhouse-gas-emissions

DATA NEVER SLEEPS 6.0. (n.d.). Retrieved from DOMO: https://www.domo.com/assets/downloads/18_domo_data-never-sleeps-6+verticals.pdf*DRAWING FROM DATA*. (n.d.). Retrieved from drawingfromdata.com: https://www.drawingfromdata.com/how-to-rotate-axis-labels-in-seaborn-and-matplotlib

E, K., & Ratanamahatana, C. (2004, April). Making time-series classification more accurate using learned constraints. *In Proceedings of the 2004 SIAM international conference on data mining* (pp. 11-22). Society for Industrial and Applied Mathematics.

Eddy, J. (2018, May 11). *Forecasting with Neural Networks - An Introduction to Sequence-to-Sequence Modeling Of Time Series*. Retrieved from jeddy92.github.io: https://jeddy92.github.io/JEddy92.github.io/ts_seq2seq_intro/
Hamilton, J. (1994). *Time series analysis.* New Jersey: Princeton.

Hees, J. (n.d.). *SciPy Hierarchical Clustering and Dendrogram Tutorial*. Retrieved from Jörn's Blog: https://joernhees.de/blog/2015/08/26/scipy-hierarchical-clustering-and-dendrogram-tutorial/

Kaggle. (n.d.). *brief-insight-on-web-traffic-time-series*. Retrieved from Kaggle.com: https://www.kaggle.com/dextrousjinx/brief-insight-on-web-traffic-time-series

Kaggle. (n.d.). *ensemble-of-arima-and-lstm-model-for-wiki-pages*. Retrieved from Kaggle.com: https://www.kaggle.com/screech/ensemble-of-arima-and-lstm-model-for-wiki-pages

Kaggle. (n.d.). *forecasting-web-traffic-with-prophet-in-python*. Retrieved from Kaggle.com: https://www.kaggle.com/sudosudoohio/forecasting-web-traffic-with-prophet-in-python

Kaggle. (n.d.). *predictive-analysis-with-different-approaches*. Retrieved from Kaggle.com: https://www.kaggle.com/zoupet/predictive-analysis-with-different-approaches

Keeling R, P. S. (2009). Atmospheric Carbon Dioxide Record from Mauna Loa. *Carbon Dioxide Information Analysis Center (CDIAC) Datasets*, doi:10.3334/cdiac/atg.035 .

Kotu, V., & Deshpande, B. (2015). Predictive Analytics and Data Mining.

Laurinec, P. (2019, 02 03). *Multiple Data (Time Series) Streams Clustering*. Retrieved from petolau.github.io: https://petolau.github.io/Multiple-data-streams-clustering-in-r/

Minnaar, A. (n.d.). *Time Series Classification and Clustering with Python*. Retrieved from alexminnaar.com: http://alexminnaar.com/2014/04/16/Time-Series-Classification-and-Clustering-with-Python.html

SciPy. (n.d.). *scipy.cluster.hierarchy.fcluster*. Retrieved from SciPy.org: https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.fcluster.html

SciPy.org. (n.d.). *scipy.cluster.hierarchy.linkage*. Retrieved from SciPy: https://docs.scipy.org/doc/scipy-0.18.0/reference/generated/scipy.cluster.hierarchy.linkage.html#scipy.cluster.hierarchy.linkage

Sharma, P. (2019, May 27). *A Beginner's Guide to Hierarchical Clustering and how to Perform it in python*. Retrieved from analyticsvidhya.com: https://www.analyticsvidhya.com/blog/2019/05/beginners-guide-hierarchical-clustering/

Shumway, R. H. (2017). *Time series analysis and its applications: with R examples.* Springer.

SINGH, A. (2018, September 27). *A Multivariate Time Series Guide to Forecasting and Modeling (with Python codes)*. Retrieved from analyticsvidhya.com: https://www.analyticsvidhya.com/blog/2018/09/multivariate-time-series-guide-forecasting-modeling-python-codes/

Sobolewska, E. (n.d.). *Dynamic Time Warping (DTW) as a mean to cluster time series*. Retrieved from rpubs.com: https://rpubs.com/esobolewska/dtw-time-series

Srivastava, P. (2917, December 10). *Essentials of Deep Learning : Introduction to Long Short Term Memory*. Retrieved from analyticsvidhya.com: https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/

StackExchange. (n.d.). *clustering multivariate time-series datasets*. Retrieved from Datascience.Stackexchange:

https://datascience.stackexchange.com/questions/19040/clustering-multivariate-time-series-datasets/19103

StackExchange. (n.d.). *hierarchical-clustering-of-time-series-in-python-scipy-numpy-pandas*. Retrieved from Stats.StackExchange: https://stackoverflow.com/questions/34940808/hierarchical-clustering-of-time-series-in-python-scipy-numpy-pandas

StackExchange. (n.d.). *How to deal with time series which change in seasonality or other patterns?* Retrieved from StackExchange.com: https://datascience.stackexchange.com/questions/3738/how-to-deal-with-time-series-which-change-in-seasonality-or-other-patterns/3764#3764

StackExchange. (n.d.). *How to interpret the dendrogram of a hierarchical cluster analysis*. Retrieved from Stats.StackExchange: https://stats.stackexchange.com/questions/82326/how-to-interpret-the-dendrogram-of-a-hierarchical-cluster-analysis

Stock, J. (2001). *International Encyclopedia of the Social & Behavioral Sciences.*

Taylor, S. J., & Letham, B. (n.d.). *Prophet: forecasting at scale*. Retrieved from https://research.fb.com/prophet-forecasting-at-scale/

Wikipedia. (n.d.). *Time series Analysis*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Time_series

ymlai87416. (n.d.). *web-traffic-time-series-forecast-with-4-model*. Retrieved from Kaggle.com: https://www.kaggle.com/ymlai87416/web-traffic-time-series-forecast-with-4-model

Yuan, Y., Ries, L., Petermeier, H., Trickl, T., Leuchner, M., Couret, C., . . . Menzel, A. (2019). On the diurnal, weekly, and seasonal cycles and annual trends in atmospheric CO2 at Mount Zugspitze, Germany, during 1981–2016. *Atmospheric Chemistry and Physics*, 19(2):999–1012. doi:10.5194/acp-19-999-2019.