News Comparison

Leyen Qian
University of Colorado at Boulder
CSCI 4502-100
jiqi2818@colorado.edu

Ziyue Guo
University of Colorado at Boulder
CSCI 4502-100
zigu7510@colorado.edu

Guofeng Deng
University of Colorado at Boulder
CSCI 4502-100B
gude8997@colorado.edu

On my honor, as a University of Colorado Boulder student,
I have neither given nor received unauthorized assistance.

1 Introduction

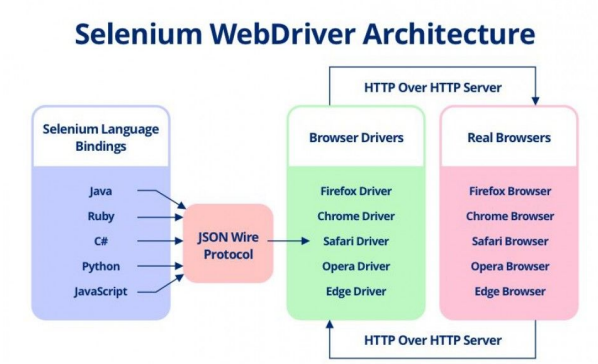
Media usually is the main approach of propaganda; using different words on report may have distinct effects on public opinions, especially when the majority of the media organizations choose to use the same words and sentences in their reports of certain affairs. The project we are going to develop is "News Comparison." In general, we collect qualified news from several media institutions' website through automation test framework. Then all information will be passed to specific module for frequent itemsets calculation, and find the similarity of the frequent itemsets in different kind of news.

One of the challenges we could meet is those websites may detect web crawling behaviors and stop providing services, like refusing connection or requiring verification. There are several solutions, for instance, we can purchase a VPN accounts, but that would be costly. Our solution is using IP address of CU, because IP addresses from large-scale organizations usually have higher tolerance for not being blocked. Furthermore, a User Agents dataset will also be applied in the news retrieving process and it is necessary for large-scale news retrieving. This dataset includes around 18,000 mobile User Agents, while it is only a fraction of the online version that has more than 4 millions of items.

There are several ways to capture data online, one popular way is sending HTTP request to the server directly, which is the most efficient method we are able to use currently. However, this method has some limitations; such as avoiding rendering detection or cannot execute Javascript on the client side. Therefore, using browser in online data retrieving has become a necessity. In order to manipulate the browsing behaviours; we need to use automation tools, selenium; it sends command to browser driver though JSON protocol. Then, the driver control the browser directly by calling public interface of browser.

In actual operation, we need to determine the relation between different page elements; For example, BBC News uses the class name 'nw-o-link' to label links to pages with different news type, while the Article and title parts are enclosed by the DIV with class name 'story-body.' Besides all that above, we also need to inject Javascript command into browser to determine when the whole DOM object is being loaded completely. This is because some websites use ajax technology to dynamically load the newest articles, which may cause an exception when we are trying to

locate web elements.



src: https://hackr.io/blog/complete-guide-selenium-webdriver

After getting the desired articles, we have to perform a preliminary data processing; Normalizing non-English alphabet to English, transforming Naïve to Naive; Converting uppercase letter to lowercase; And delete all punctuations.

2 Related work

In this section, we briefly introduce some papers with the topics of analyzing some news related information. We get some insight from theses papers, and may take some of their methods as references.

2.1 DEFEND: Explainable Fake News Detection [1]

Use some machine learning algorithms to detect the fake news. Especially, they use an explainable machine learning method so that we can also understand why such news are considered to be fake. In detail, they use the RNN (Recurrent Neural Network) with a method called GRU (Gated Recurrent Unit) to encode the words and sentences. They also introduce a method called Sentence-Comment Co-attention which can include the information of comments to help detect fake news.

2.2 NPA: Neural News Recommendation with Personalized Attention [2]

They build a news recommendation system based on a news representation model and a user representation model. Specifically, they use a CNN network (Convolutional Neural Network) to learn the contextual representations of news titles.

The ways the previous two papers analyze the texts of the news are interesting, and we can get some insight on analyzing the text of an article from their method.

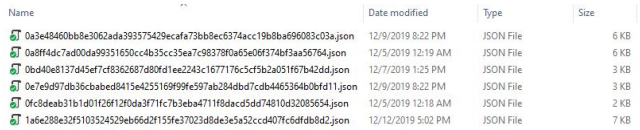
2.3 MediaRank: Computational Ranking of Online News Sources [3]

They build a website called MediaRank, which have ranks on a large amount of news sources around the world. One of the significant contributions in their paper is that they propose a new computational methods for measuring influence and bottomline pressure. This gives us a new idea on analyzing the news articles. However, we may be more focused on the text analyzing, instead of using the other relevant information. Specifically, our project focus more on the information of the text in each news article, and probably the relation between different articles.

3 Proposed work

The size and real-time of the Dataset is important to accurate analysis. Therefore, it is a necessity that running crawling program daily; Under strict filter mode, we are able to get around 1000 articles after a few days of retrieving. As the increase in the article database, the relation among words, and the connection of certain words and articles under a specific News category become even obvious; For instance, words “trump”, “tariff” and “house” have more chances that occur in the same article, which seems obviously for most people. we will reveal more results later. Some News may appear in different categories at the same time, for example, a political News may appear in business categories just because this News describe a newly enacted act that may take effects on economic growth. If we switch the filter mode to “lazy,” we should able to crawl more articles than what we have got currently by at least 20% (figure here may not accurate due to the change of News from the data source).

Articles will be stored as json files initially; For each file, there will be unique sha256 value as the proof of the existence of an article from the specific data source, which is the key to implement incremental data retrieving.



Name	Date modified	Type	Size
0a3e48460bb8e3062ada393575429ecafa73bb8ec6374acc19b8ba696083c03a.json	12/9/2019 8:22 PM	JSON File	6 KB
0a8ff4dc7ad00da99351650cc4b35cc35ea7c98378f0a65e06f374bf3aa56764.json	12/5/2019 12:19 AM	JSON File	6 KB
0bd40e8137d45ef7cf8362687d80fd1ee2243c1677176c5cf5b2a051f67b42dd.json	12/7/2019 1:25 PM	JSON File	3 KB
0e7e9d97db36cbabed8415e4255169f99fe597ab284dbd7cdb4465364b0bfd11.json	12/9/2019 8:22 PM	JSON File	3 KB
0fc8deab31b1d01f26f12f0da3f71fc7b3eba4711f8dacd5dd74810d32085654.json	12/5/2019 12:18 AM	JSON File	2 KB
1a6e288e32f5103524529eb66d2f155fe37023d8de3e5a52ccd407fc6dfdb8d2.json	12/12/2019 5:02 PM	JSON File	7 KB

In detail, each json file contains three fields “title”, “link” and “article;” those are the most significant part for data analysis.

```
{
    "Title": "World trade without rules? US shuts down WTO appeals court",
    "Link": "https://abcnews.go.com/Business/wireStory/world-trade-rules-us-shuts-wto-appeals-court-67605603",
    "Article": "\nGlobal commerce will lose its ultimate umpire Tuesday, leaving countries unable to reach a final resolution
}
```

3.1 **Dataset**

Our dataset is the words of articles and published by the news websites such as BBC news [4], NBC news [5], ABC news [6], China Daily [7] and UserAgent [8]. The automation test framework, selenium, is the core component of new retrieving. It was used in my previous project, search engine optimization. With selenium, we are able to crawl data from web page in the usual manner; this method is effective when the website is sensitive to the web spider. Although the idea behind two projects are different, we still able to find components that shares in common. But the basic code will be refactored, and also adding necessary exception handling for better performance and stability.

3.2 **Subtasks**

3.2.0 **Frequent Words in Life**

The frequencies of all words are different in real life. For example, the word "the" is much more frequent than the word "university". In any article, we can expect that the number of "the" is more than the number of "university". The frequencies of some words are almost zero in real life, such as "dioxide". The word "dioxide" may occur in the reports of population, but the frequency is expected to be still less than that of "the". Therefore, we have to weight the frequencies of words in news by the frequencies of words in life. Fortunately, there are a lot of statistics about English words frequency. We will choose ⅓ million most frequent English words on the web [9] as a reference.

We load the data of "⅓ million most frequent English words on the web" by using Pandas Dataframe in Python. The result is similar as my prediction. The frequency of the word "the" is 0.039338, while the frequency of university is only 0.0005294, which is smaller by a factor of 74. Since there are ⅓ million words in this dataset, most of the word is not frequent. So we set the frequency support threshold s = 2.5e-7, and delete data of all words with frequency lower than 2.5e-7.

3.2.1 **Weighted Frequencies of Words in News**

Generally, the word choices in news is more serious than the word choices in life. The frequency of casual words is expected to be much lower in news than in real life. First, we calculate the frequencies of words in our dataset. For each frequency of words in the list of frequencies, divide it by the frequency of the same word in real life, and define the generated ratio as **weighted frequency**. We will find the weighted frequencies of words in news, and then find the weighted frequencies of words in different topics, such as sports news, finance news, etc.

to	0.942	by	3.758
police	133.630	protester	48751.249
protesters	3794.020	hong	334.867
gas	79.968	were	15.757
on	1.525	ho	718.503
at	2.517	on	2.398
tear	1020.278	that	2.645
death	57.272	wan	1361.164
were	8.351	live	43.337
shopping	23.351	and	0.553
over	10.378	fired	895.697
student	33.192	two	16.299
fired	474.705	sai	4530.851
windows	22.624	as	2.401
arrest	498.156	morning	100.959

The pictures above shows the weighted frequencies of some words in 2 different news related to Hongkong. In both articles, the word "protester(s)" appears thousands times more frequent than a normal article on the web. And the word "fired" is hundreds times more frequent. So these two words may be very important and frequent in the news related to Hongkong.

A problem to find the weighted frequency is, some of the words are miss-spelled. These words should have frequency 0.

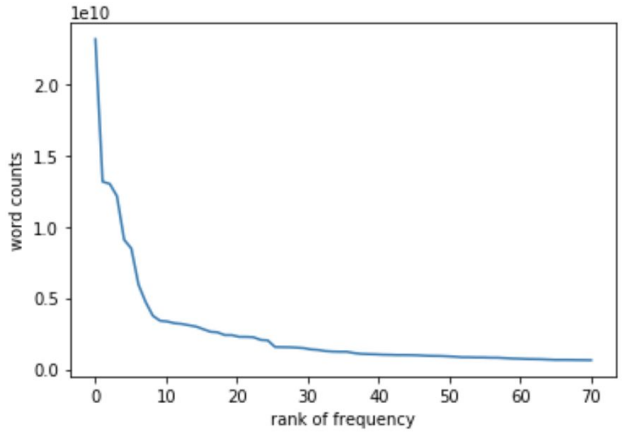
antigovernment	0.000
during	9.238
northwest	93.739
men	10.953
station	30.802
up	2.297
tsuen	0.000

However, some of the words may be distinguished to be miss-spelled because of the punctuation between the word. For example: "anti-government". Since we remove all punctuation, the character '-' is deleted. Then the word becomes "antigovernment", which we can not match the data in our dataset of frequency of all words. We still need to think about the solution.

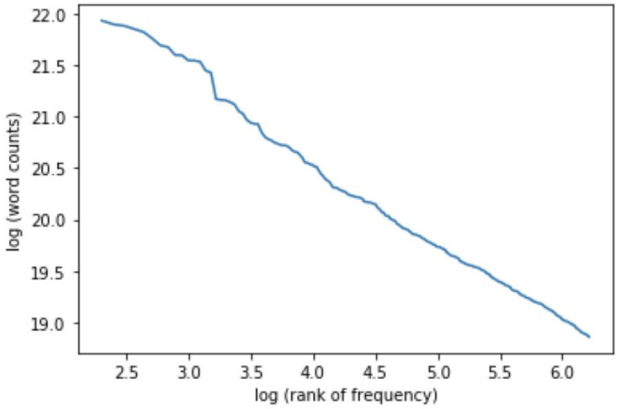
3.2.2 **Market Basket Analysis**

Frequent Words selection

Another method to find the frequent words is the market basket model. We will set an appropriate support threshold. Since some of the words are very common, we have to remove them so that they while not affect our result.

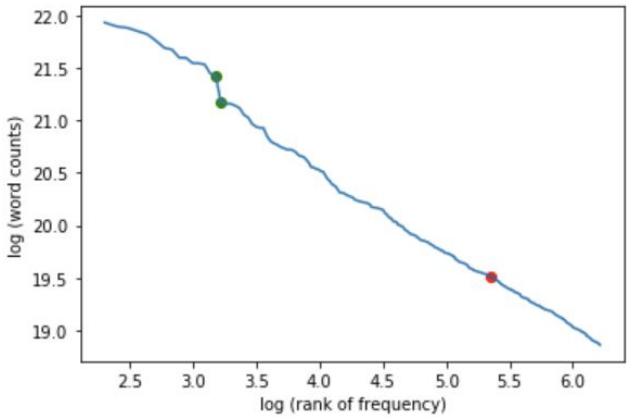


The plot above shows the relation between word counts (frequency) and ranks of frequency (No. 1 ~ 70). However, the plot is non-linear, so it will be difficult for us to determine a good standard for common words. So we make another plot, which is in log-log scale.



This time, we chose the rank of frequency from No. 5 to 500, and we get an almost linear result. the curve decreases smoothly at the beginning, and then meets a cliff between rank = 25 and 26. Then the curve decreases linearly after rank = 26. So we will regard the top 25 frequent words as common words, and remove them. These words are: “the”, “of”, “and”, “to”, “a”, “in”, “for”, “is”, “on”, “that”, “by”, “this”, “with”, “i”, “you”, “it”, “not”, “or”, “be”, “are”, “from”, “at”, “as”, “your”, and “all”.

However, when we tried to find the frequent item set, there are still too many common words, so we finally enlarge the size of common word set to 210.

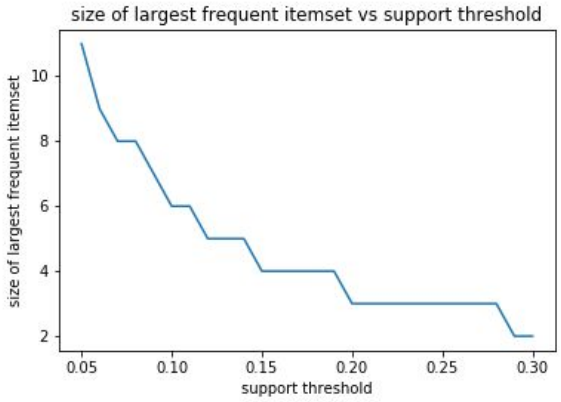


Frequent Itemsets Analysis

We then tried to analyze the news articles based on frequent itemsets analysis. We choose each news article as a basket and the words appearing in the corresponding news article as the items in each basket. In our case, the frequent item with size 1 would be the single word that appears frequently in the selected set of news articles, and the frequent itemsets with size larger than 1 would be the combination of words that appear together frequently in the selected set of news articles. By investigating these frequent words or combination of words, we can get a sense of the most important information among the selected set of news articles. For example, if we choose the set of news articles to be in the same month, then by looking for the frequent word or frequent combination of words, we can get a quick view of the biggest, and mostly mentioned information happened in the corresponding month. More specifically, if we choose the set of news articles to be in the same topic, say politics, happened in the same month, then by looking for the frequent word or frequent combination of words, we can get a sense of the mostly mentioned information in politics area during the selected month, and probably the most important information in the chosen area during the selected month.

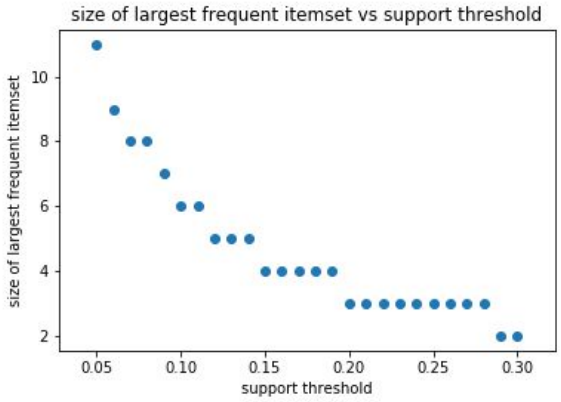
We tried to find the frequent itemsets using the related basket-item algorithms. In our code, we chose to use apriori algorithm to find the frequent itemsets. We first used the function in mlxtend package to encode our crawled news data, and then used the apriori function in mlxtend to find the frequent itemsets. We tried to use different minimum support thresholds and see the performance. However, if the minimum support threshold is too small, then there would be too many frequent itemsets, which would cause a considerably long runtime. Also, if the minimum support thresholds are too big, then only a few words or combination of words would be treated as frequent, and few information would be mined. Thus, we did some analysis on the choice of the minimum support threshold.

The data we are using in the analysis here is the news articles during several days in the beginning of December 2019. The topics include Business, Politics, and some general U.S. domestic news. We made several trails on the selection of the minimum support threshold, and we found that when the size of the largest frequent itemsets is beyond 11, then the runtime would be too long. Thus, we set the biggest size that the largest frequent itemsets can reach is 11. We then chose a set of the potential support threshold, and searched for the corresponding size of the largest frequent itemsets. The corresponding graph is as shown below:



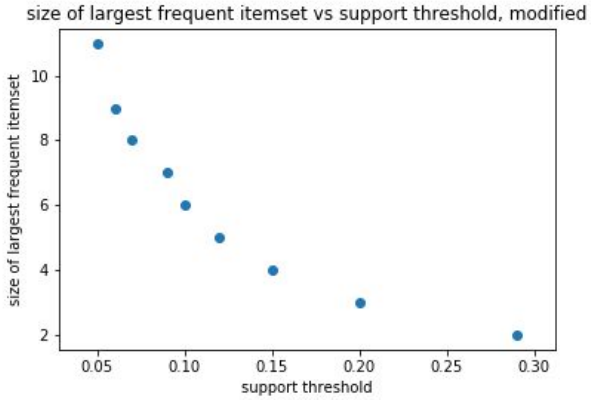
In this graph, the x axis is the minimum support threshold, and the y axis is the corresponding size of the largest frequent itemset. The choice of our minimum support threshold is smaller than 0.3, since the case would be trivial if we choose some larger size. Also, the smallest choice is about 0.5, since the runtime gets too long if smaller minimum support threshold are chosen.

We can also see this graph in a more intuitive view in the scatter graph.



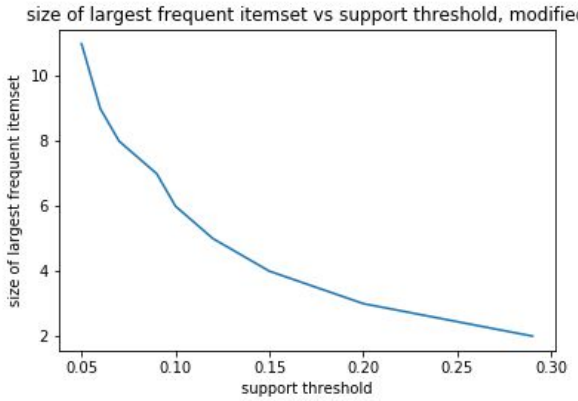
This is the scatter version of the graph. We can see that the as the support threshold gets larger, the size of the largest frequent itemset decreases in a smaller rate.

We then choose to only investigate the turning points: the point that the size of the largest frequent itemset change with the increase of the support threshold. The resulting graph is as follows:

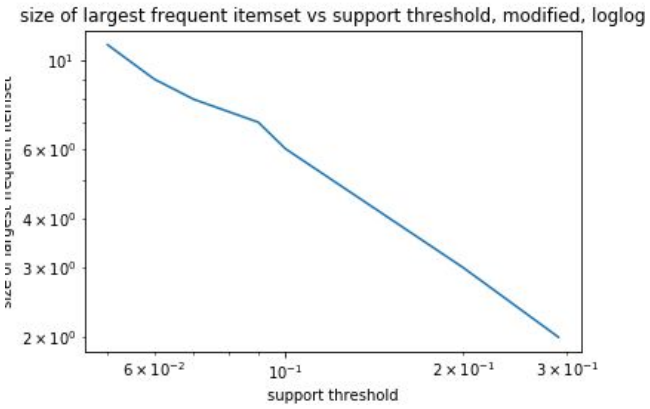


This graph shows the trend of the size of the largest frequent itemset with respect to the support threshold in a better way, comparing to the previous one.

And if we plot it with the points connected, we get the following graph:



We can see that the resulting graph is showing a trend similar to some inverse exponential relation between the size of the largest frequent itemset and the support threshold. Thus we tried to draw a loglog plot corresponding to these data:



And as we expected, it shows a trend which is very close to a linear relation.

After the analysis of the choice of minimum support threshold, we then investigated the resulting frequent combination of words. One of the combinations of words (frequent itemset) with a size of 11 is:

{'inquiry', 'democrats', 'trumps', 'presidents', 'congress', 'trump', 'political', 'donald', 'president', 'impeachment', 'house'}

Interestingly, we the word-pair of ‘donald’ and ‘trump’ appears a lot in our data. It is not surprising since the name of the president is always one of the most important and most frequently mentioned word-pair.

Other words like ‘congress’, inquiry', or 'impeachment' are also the important words in the Politics and Business area.

3.2.3 **k-Shingle Analysis**

We are also interested in the frequent phrase. However, a phrase could be constituted by different number of words. The phrase formed by successive k words may be very popular in news, So we want to find the most frequent length k phrase using Jaccard jaccard similarity or minhash similarity.

We have implemented the code for extracting the k-shingle from a given news article for arbitrary k , and then compute the similarity between two news articles based on the extracted k-shingle information. We then apply this method to some sample news articles. We found that the similarity between two articles within the similar topics indeed have a higher result, comparing those news articles pairs in different topics. However, when actually applying this method to some sample news article, we found that the length of the news articles may be a problem when trying to apply the k-shingle method with a larger k. Most of the news article is short. Thus we can hardly find the same k-shingle from two different news articles when k gets large.

The sample news articles we have chosen is 5 news articles about sports (more specific, NBA) and 2 news articles about politics (more specific, Hong Kong). We investigated the similarity between the 2 politics news articles, the similarity between the 5 sports news articles, and the similarity between one politics news article and one sports article.

When K = 2, in order words, we construct each shingle with 2 consecutive words in a news article, the corresponding similarities are shown below:

```
K = 2
Similarity between:

between politics:
HK1 and HK2: 0.025189

between sports:
NBA1 and NBA2: 0.046369
NBA1 and NBA3: 0.036379
NBA1 and NBA4: 0.035907
NBA1 and NBA5: 0.042791
NBA2 and NBA5: 0.036895

between one politics and one sports
HK1 and NBA1: 0.008867
HK2 and NBA1: 0.009321
```

By observing these values, we can find that the similarities between the news articles in the same topics are significantly larger than the similarities between the news articles in different topics. This method works well here when K = 2.

However, when K = 1, namely we construct each shingle with 1 single word in a news article, the corresponding similarities are shown below:

```
K = 1
Similarity between:

between politics:
HK1 and HK2: 0.133188

between sports:
NBA1 and NBA2: 0.196998
NBA1 and NBA3: 0.19708
NBA1 and NBA4: 0.198842
NBA1 and NBA5: 0.187377
NBA2 and NBA5: 0.201331

between one politics and one sports
HK1 and NBA1: 0.078853
HK2 and NBA1: 0.112532
```

This time, we can still see a difference between the value for the same topics and for different topics, but not as significant as the difference when K = 2. For example, the value for HK1 and HK2 is very close to the value for HK2 and NBA1, while these two values should be differed significantly in order to distinguish the relationship between two news articles (whether they are in the same topics or not).

Then we investigate for some larger K. The result for K = 3 is shown below:

```
K = 3
Similarity between:

between politics:
HK1 and HK2: 0.006928

between sports:
NBA1 and NBA2: 0.010811
NBA1 and NBA3: 0.009105
NBA1 and NBA4: 0.006457
NBA1 and NBA5: 0.014274
NBA2 and NBA5: 0.009662

between one politics and one sports
HK1 and NBA1: 0.0
HK2 and NBA1: 0.0
```

When the K gets larger, we can expect the values will all get smaller. This is because as the K gets larger, each shingle will have longer length, including more words. Thus, the occurrence of a shingle become rarer. It becomes harder for two news articles to have the same shingle.

We can see that the value for the same topics become smaller, which is as expected. However, we can see the value for two different topics become 0. This is because as the K gets large, it becomes harder for two news articles with different topics to have the same shingle. In our sample news articles here, the value for news articles under different topics become 0 when K gets to be 3, and we can no longer increase K to do more analysis. As mentioned above, this is because the news articles are not long enough to apply this method.

Previous part is the analysis for the selection of K. Then we investigate the semantics from our results. We do the same thing as what we did when calculating the similarities between two news articles. Instead of only calculating the similarities, here we also print out the corresponding shingles that appears in both of the news articles we are comparing. We will start with the case when K = 2.

When K = 2, the shingles appearing in both of the Hong Kong (Politics) news articles are:

“during a”, “death of”, “executive carrie”, “between police”, “the death”, “over the”, “police and”, “extradition bill”, “on the”, “fell from”, “have been”, “from a”, “that began”, “protesters in”, “chief executive”, “who fell”, “and the”, “in the”, “across hong”, “hong kong”

The appearance of these 2-shingles (phrases) make sense since the two politics news articles are about the recent Hong Kong events.

The shingles appearing in both of the NBA (sports) news articles are: (when comparing NBA1 and NBA2)

'left to', 'just seconds', 'the lead', 'games of', 'have to', 'pair of', 'to give', 'from the', 'by in', 'a lot', 's in', 'with a', 'after a', 'had points', 'and a', 'i thought', 'the second', 'seconds left', 'to the', 'the game', 'pm mtassociated', 'over the', 'give it', 'it to', 'on the', 'in a', 'rebounds and', 'the third', 'to help', 'loss in', 'of the', 'including a', 'first half', 'a pair', 'one of', 'at home', 'win over', 'in the', 'of s', 'points in', 'points and', 'the stretch', 'mtassociated pressfacebooktwitterfacebook', 'we didnt', 'they were', 'into the', 'with the', 'said i', 'a with', 'the first', 'hit a', 'the denver', 'down the'

The appearance of these 2-shingles (phrases) also make sense given that most of these phrases are some basketball terminology.

However, the shingles appearing in one sports news articles and one politics news articles are listed below:

'during a', 'to the', 'over the', 'on the', 'in a', 'of the', 'who was', 'but the', 'in the'

We can see that most of these 2-shingles (phrases) are some commonly used phrases.

By investigating these results, we see that the k-shingle method for K =2 works well for detecting the commonly used phrases in

news articles, both in the ones under the same topics or under the different.

Further, as the value of K gets greater, the performance of this method gets better. The meaning of the resulting k-shingles become clearer. In our sample case, when K = 3:

The shingles appearing in both of the Hong Kong (Politics) news articles are:

'between police and', 'fell from a', 'who fell from', 'across hong kong', 'chief executive carrie', 'protesters in the'

We can see that almost all of the 3-shingles (phrases) appearing in both of the two news articles are related to the main content of these two news articles.

Also the shingles appearing in both of the NBA (sports) articles are: (when comparing NBA1 and NBA3)

'in the third', 'a lot of', 'in the fourth', 'of his points', 'in the first', 'points in the', 'the first half', 'said of the', 'had points and', 'victory over the', 'a pair of', 'his points in'

Again, almost all of the 3-shingles (phrases) appearing here are the basketball terminology.

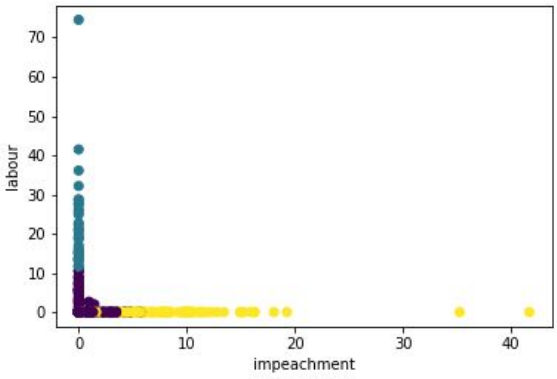
We can see here as the value of K gets bigger, the resulting k-shingles have better semantics performance.

However, the maximum valid value of K depends on the length of the news articles. If the length of the news articles can somehow get bigger, we can then apply the k-shingle method with some bigger K, and hopefully gets some better performance

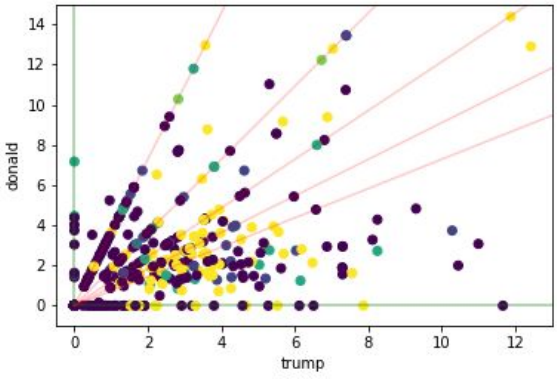
3.2.5 **K-Means Clustering of Articles**

At the initial, we can define k = number of types of news, and for each type, we randomly select an article to be the initial centroid of the cluster. We will try both Euclidean distance and Jaccard distance and choose the better one. And finally, we can use the clusters to do some further analysis. For example, we may use the similar methods as above to find some frequent items (words, phrases, sentences) of each cluster. Such frequent items may show some specific features of each cluster.

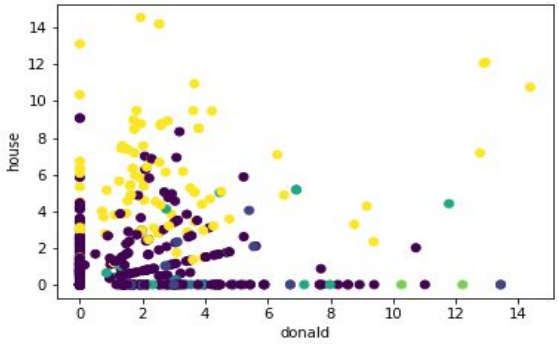
We use our results in frequent 11 item basket of our 9 data sets, and select 10 words from them. Some of the frequent words always come together, such as “president” and “trump” (president trump), “dow” and “jones” (dow jones). The words in pairs or sets are not independent. It is very difficult to show the cluster of 10 dimensions data using a plot. If the two axes we choose are linearly independent, the plot of points should be in “L-shape”.

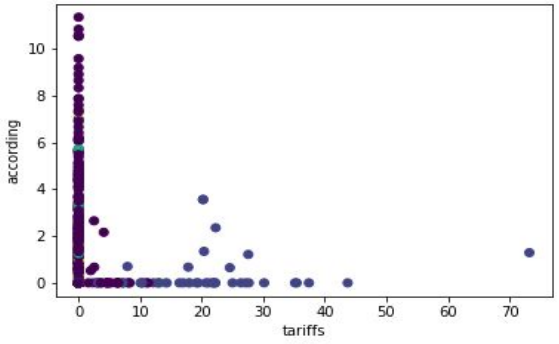


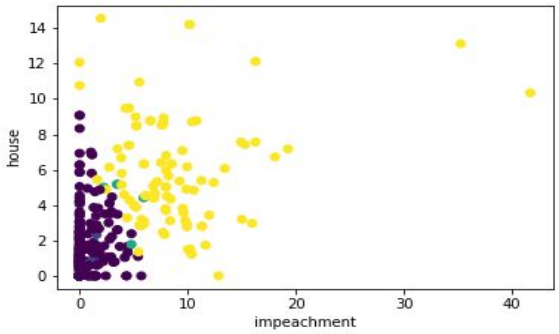
The plot above is the normalized frequency of words “labour” versus “impeachment”. weighted frequency will be so high once the word occurs in the articles. So we normalize the weighted frequency by dividing the average of weighted frequency. We can find that the y axis of these plot has larger magnitude, because the word “labour” is not as frequent as “house”. The normalized frequency of this word in many articles is 0, so the magnitude of other non-zero numbers have to be larger to make sure the average normalized frequency is 1. Since the word “labour” and “impeachment” are independent, when one of them occurs, the probability of the other occurs is very low, so only a few articles are away from the axes. However, if two words are highly related (“donald” and “trump”), the distribution of points in these two axis will spread in the space. We can even find some linear relationship of the scatters (read lines). Many of the points are on the lines donald = 3.64*trump/i, where i is equal to 1~5.

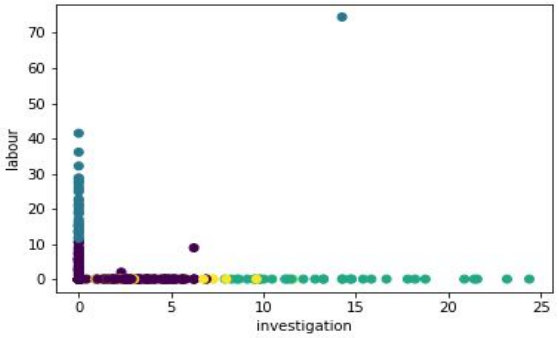


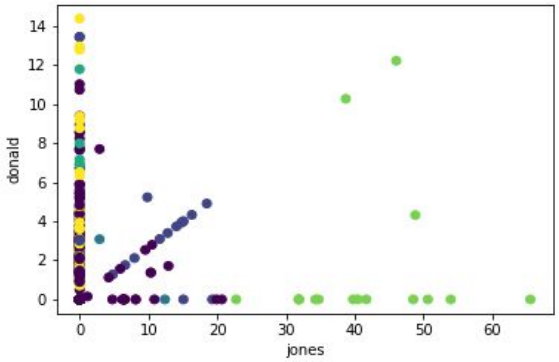
Linearly dependent words may interrupt the result, but it may be difficult to find the k-means cluster in high dimension. finally we choose these 10 words as our axis:
kword = ['trade', 'tariffs', 'jones', 'trump', 'impeachment', 'donald', 'house', 'investigation', 'according', 'labour']
Here are some of the plots to show the relations between words;

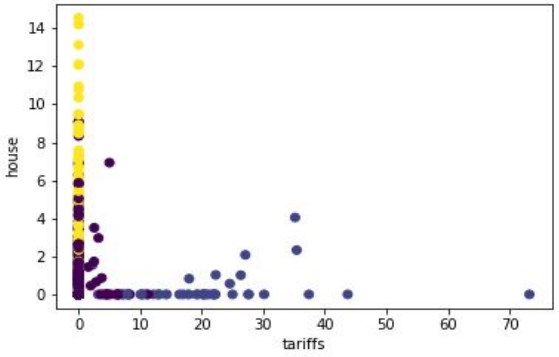

house
donald

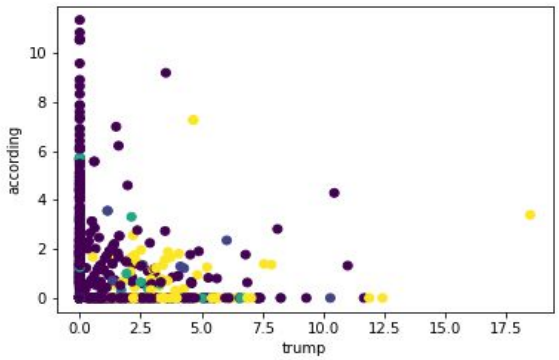

according
tariffs

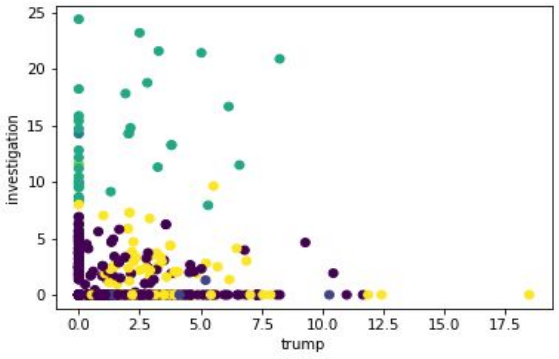

house
impeachment

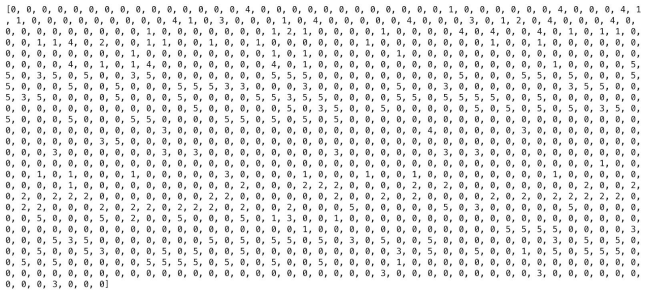

labour
investigation


donald
jones


house
tariffs

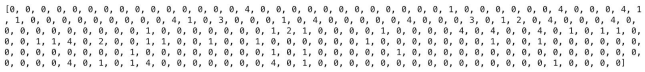

according
trump


investigation
trump

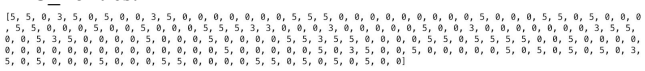


The plot shows above is the result of our k-means clustering when k = 6. The result is not ideal, due to the very high dimension. There are total 1031 articles in our dataset, however, 831 of them are in a huge center cluster (when k = 3, the number of points in center cluster is 971, 94% of the data)

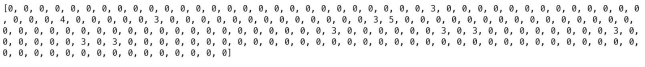
ABC_Business:



ABC_Politics:

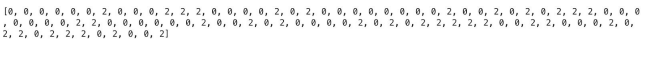


ABC_US:



BBC_Business:

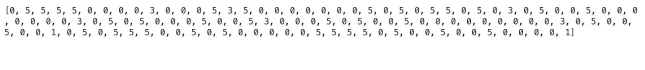
BBC_Politics:



BBC_US_Canada:

CNN_Business:

CNN_Politics:



CNN_US:

In our dataset, ABC Politics and CNN Politics are both consisted by large amount of elements in cluster 3 and cluster 5. Whereas BBC Politics is contributed by cluster 2. ABC Business, BBC Business and CNN Business all have large amount cluster 1 members. And for all articles relate to the US, they are all consisted by huge amount of 0, the word choice may not be very unique.

Consequently, The word choice of ABC Politics and CNN Politics are very similar, but that of BBC Politics is different. The business articles in all three websites have similar word choices.

3.3 **Workload**

There are three members in our group, and all of us are undergraduate students. The works above are sufficient for us. And the works are also feasible, because the dataset is accessible for us. We plan to finish one subtask per week.

4 **Evaluations**

For the frequent words and k-shingles (phrases, sentences), there may not be valid mathematical ways to evaluate our results. However, we can directly observe the frequent words and k-shingles, and these frequent items may make some sense since they are extracted from the news articles (probably from some specific topics).

For the clustering, we can compare our clustering results with the actual classification of the news article, which is already labeled when we are collecting the data. However, the result of such comparison may only be a reference, instead of a way to evaluate the accuracy, since we are looking forward to discovering some unknown relationship between news articles. Furthermore, the frequent items we extract from each cluster may be representative to the information of the cluster.

Since the information we are mining is hard to express in numerical way, there are few mathematical evaluation to compare our results with other methods considered as baselines.

5 **Milestones**

Week 7: Project Proposal

Week 8: Retrieve Data

Week 9: Weighted Frequencies

Week 10: Market Basket Analysis

Week 11: k-Shingle Analysis

Week 12: Project Checkpoint

Progress report

Week 13: K-Means Clustering

Week 14: K-Means Clustering (Jaccard)

Week 15: Report Finished/ Presentation Prepared

Week 16: Project Finished

Final project report

Final project presentation

REFERENCES

[1] Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, and Huan Liu. 2019. dEFEND: Explainable Fake News Detection. In The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'19), August 4–8, 2019, Anchorage, AK, USA. ACM, New York, NY, USA,11 pages. https://doi.org/10.1145/3292500.3330935

[2] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. NPA: Neural News Recommendation with Personalized Attention. In The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330665

[3] Junting Ye and Steven Skiena. 2019. MediaRank: Computational Ranking of Online News Sources. In The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330751

[4] BBC News, https://www.bbc.com/news

[5] CNN News https://www.cnn.com

[6] ABC News https://abcnews.go.com/

[7] China Daily http://global.chinadaily.com.cn/

[8] UserAgent https://developers.whatismybrowser.com/useragents/explore/

[9] Rachael Tatman. 2017. ⅓ million most frequent English words on the web. Retrieved from https://www.kaggle.com/rtatman/english-word-frequency