Topic Modeling on BBC and Cointelegraph Headlines – BSHDS4 CA1

Jordan O'Donovan
School of Computing
National College of Ireland
Dublin, Ireland
x19372016@student.ncirl.ie

***Abstract*— We as humans like to have information as condensed as possible, that's why many people get all of the information they need from an article's headline. I will use Latent Dirichlet Allocation, Latent Semantic Analysis and Hierarchical Dirichlet Process to further reduce the length of headlines and group by their key words.**

***Keywords*—LDA, LSA, HDP, Topic Modeling, Coherence Score**

I. Introduction

The British Broadcasting Corporation (BBC) was founded one-hundred years and a month ago (originally as British Broadcasting Company) and is the national broadcaster of the United Kingdom (UK). The BBC is the world's oldest national broadcaster and currently has over twenty-two thousand employees, making them the largest broadcaster by number of employees. They have a budget of over four and a half billion British Pounds and in 2021 had revenue of just over five billion Pounds.

Cointelegraph is a solely online news publication founded in New York, 2013 with the aim of being the leading independent digital media resource covering a wide range of news on blockchain technology, crypto assets and emerging fintech trends [1]. They receive and average of twenty-three million page views per month from an average of eight million unique visitors, with almost half of their traffic coming from North and South America, as well as over a quarter from Europe and almost a fifth from Asia [2].

For this assignment I will be performing cleaning and pre-processing on thousands of headlines from these websites, before carrying out my exploratory data analysis (EDA) on the datasets. I will then create my models to perform text analytics – namely topic modeling on these headlines.

II. The Corpora

A. BBC

This dataset is a subset of a much larger dataset of headlines from bbc.co.uk/news. Whereas the original was created by Crawl Feeds and contains 1.17 million headlines, this subset contains 15,835 instances, almost all of which are headlines from 2010. This dataset contains thirteen columns: Tags, which contains some keywords from the article, Title, being the headline, News_post_date, which is the date and time the article was posted at, Raw_content which is the HTML code of the webpage, Content, the body of the article, URL, Author, Language and Category columns, an ID column to serve as a unique identifier for the webpage, a column for the region the article was written about, Short_description being the subheading for each article and finally Crawled_at, containing the datetime value for when the data for a given webpage was captured at.

B. Cointelegraph

The other dataset I will be using, contains the headlines from Cointelegraph. This dataset was also created by Crawl Feeds and contains 1,127 instances, each with nine columns which are highly similar to those of the dataset discussed previously. These columns contain the headline ("Title"), the URL for the webpage ("URL"), the datetime value for when it was originally posted ("Published_at"), the author of the news article ("Author"), the JPEG image attached to the header for each article ("Header_image"), the HTML code for the webpage ("Raw_content"), keywords to link similar articles ("Tags"), a column giving the name of the site the article was posted to ("Publisher") and finally, the datetime value for when the data was captured at ("Scraped_at").

III. Aims and Objectives

For this project I will be performing topic modelling on the headlines for the two websites. I aim to use Latent Dirichlet Allocation (LDA), Latent Semantic Analysis (LSA) and Hierarchical Dirichlet Process (HDP) to carry out this objective. I also intend to create bags of words (BoW) when pre-processing the tokens to create my term document frequency variables for each dataset. I aim to get my coherence scores as high as possible to understand the quality of my LSA, LDA and HDP models.

Due to my current computation power, I will be keeping the number of topics for each model to a somewhat low degree - probably fifteen topics.

IV. Overview of Methods Selected

A. Bag of Words

A bag-of-words is a representation of text that describes the occurrence of words within a given document. There are two metrics given by a BoW model: the vocabulary of known words, and a measure of the presence of known words. This model disregards where in the document these known words appear, instead it only wants to know if they appear or not. This is why it's referred to as a "bag", information about the order of words is not wanted thus the model discards it [3].

BoW is used as a feature extraction method for a corpus. When a BoW model is introduced to a corpus, it first creates a list of every unique word which appears. The model then creates document vectors which assign each word in a given body of text (usually a sentence) a Boolean value of 1 if they appear, a 0 if not. By default, this model does not alter the corpus thus if a given corpus contains many unique words, very large vectors will be created, with the majority of the values enclosed being 0. These are known as sparces vectors. The larger these vectors get, the higher the memory usage and computational power needed will become.

To overcome this, steps can be taken, such as raising or lowering the case of each word in a corpus, thus "Example" and "example" are not treated as being two unique words. Punctuation can also be removed although one of the first ways one would reduce the number of words in the corpus would be to remove the stop words, as they will likely not be providing any useful information in our results.

Stemming and lemmatization can also be used to reduce the vocabulary of a document. Stemming removes the first or last few characters from a word if they contain common prefixes of suffixes. "Caring" would become "Car". Lemmatization works in a similar way, although it takes the context into account, thus "Caring" may become "Care". Lemmatization requires more computational power than stemming so that it can process the context of a word.

B. *Topic Modelling*

Topic Modelling is a method of recognizing the words from a document and clustering them into word groups and similar expressions using unsupervised machine learning. Word frequency and distance between words (tokens) can be used to create these clusters [4].

C. *Latent Dirichlet Allocation*

LDA is a topic modelling approach which uses Dirichlet distribution to locate topics for each document model and words for each topic model.

LDA ignores syntactic information and treats documents as bags of words. It also assumes that all words in the document can be assigned a probability of belonging to a topic. That said, the goal of LDA is to determine the mixture of topics that a document contains [5].

LDA takes two hyperparameters: alpha and beta. If a high alpha is used, fewer topics will be assigned to each document and if a high beta is used, less words will be used to model a topic so that topics will be less similar to each other.

Term frequency-inverse document frequency (TF-IDF) may also be used instead of a BoW model.

D. *Latent Semantic Analysis*

Latent Semantic Analysis, also known as Latent Semantic Index (LSI) also uses TF-IDF or a BoW model when creating a term-document matrix. The model learns latent topics by performing a matrix decomposition on the document term matrix using Singular value decomposition (SVD) [6].

Generally, LSA requires less computational power than LDA, but it tends to produce less accurate results.

E. *Hierarchical Dirichlet Process*

Hierarchical Dirichlet Process is an extension of LDA, designed to address the cases where the number of topics is not known. It's a nonparametric Bayesian approach to clustering grouped data [7]. HDP uses a Dirichlet process for each group of data. It's an unsupervised topic model. Each topic is treated as a distribution of words in a known vocabulary. A Dirichlet process is also used to measure the uncertainty for the number of topics to use.

V. Related Work

In 2016 Zhou Tong and Haiyi Zhang used LDA to create two models, one trained on Wikipedia articles and the other on Tim Cook's Tweets [8]. They had two thousand Wikipedia articles and used topic modelling to create another model which would find the article in the dataset which was related the most to the first. One example of a result they got was that "Beamline" was most related to the article "Light", thus their models appear to be accurate.

Their model based on the tweets was much similar to what I will be attempting to do in this project. Here, they created thirty topics containing different tokens of similar subjects or categories Tim Cook tends to Tweet about. I say that this is similar to what I aim to achieve as my goal is to split the headlines into different topics which has a clear category, feeling or seriousness.

In 2020, Y. Kalepalli, S. Tasneem, P. D. Phani Teja and S. Manne also created LSA and LDA models on a BBC News dataset, but the objective of this paper was to compare the results of the two topic models mentioned to traditional machine learning algorithms, - those being K-nearest neighbours (KNN) and "NAÏVE". The results from their paper were that, unsurprisingly, both LDA and LSA were more accurate methods for topic modelling [9].

Trefor Williams and John Betak compared the performance of LSA and LDA for text mining/topic modelling the text detailing railroad accidents. From their findings, they concluded that both algorithms are effective, but as they give somewhat different results, both algorithms should be used in tandem [10].

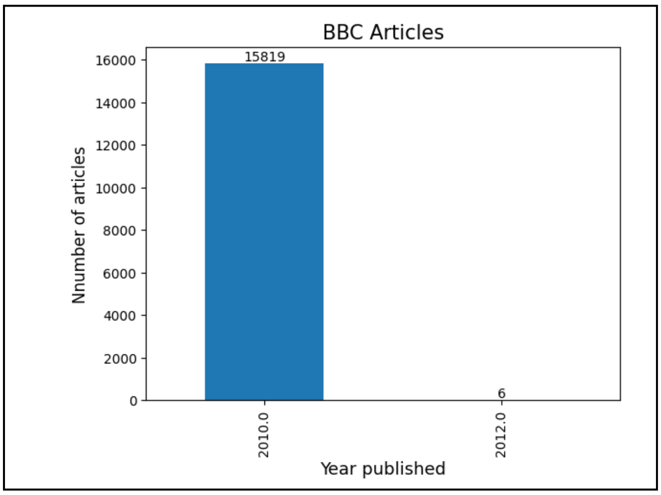
VI. Cleaning and EDA

Both datasets to be used in this assignment contain a lot of data which will not be useful for the objectives laid out previously.

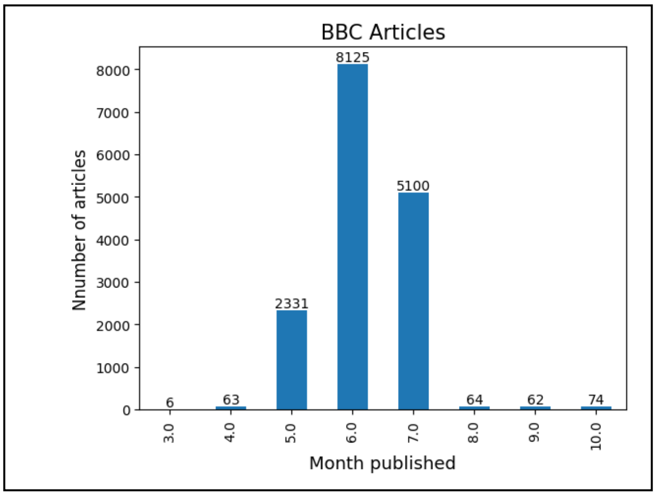
A. *BBC*

For this dataset, the first column seen, "Tags", only has data for one of its 15,835 rows thus it must be removed. 90.81% of the "Author" column is also misisng. Around 2.5% of the "Raw_content", "Content" and "Region" columns have no data. There are six headlines which have no data (0.04%). These six rows will be removed. The columns containing the HTML code, the content of the article, the URL, author, ID, language, subheading, category and "Crawled_at" will not be used with our models as they do not contain relevant information.

Four of the publication dates for the headings contain tokens instead of datetime values thus they will also be removed. Of the remaining 15,825 instances, only six of them were published in 2012, the remaining majority were posted in 2010. Two of these six were posted on the sixth of March, five seconds between eachother while the final four were posted six days later. Of the six, four of them contain "Slovak" in the headline. The other two each contain "Russia".



When the month each article in this dataset was published at, it is seen that 98.28% of them were in May, June and July. All of these come from the 2010 subset, with 62-64 each of them being from April, August and September and a final 74 from October. They only six in the dataset which occurred in March come from the 2012 subset.



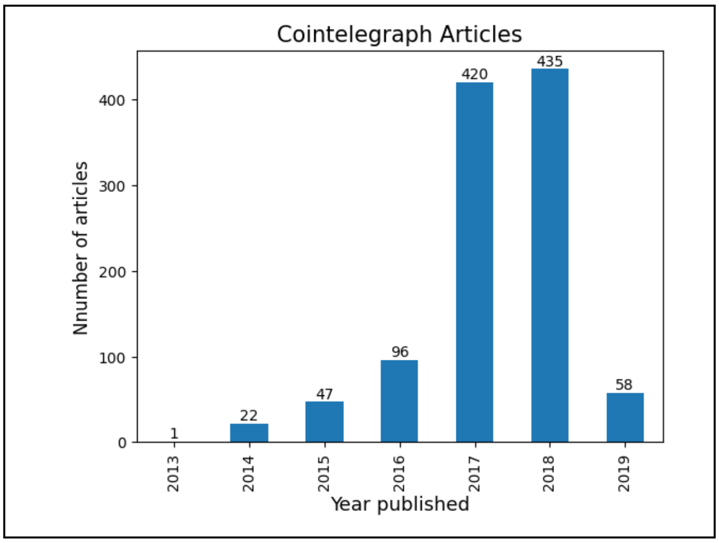
B. Cointelegraph

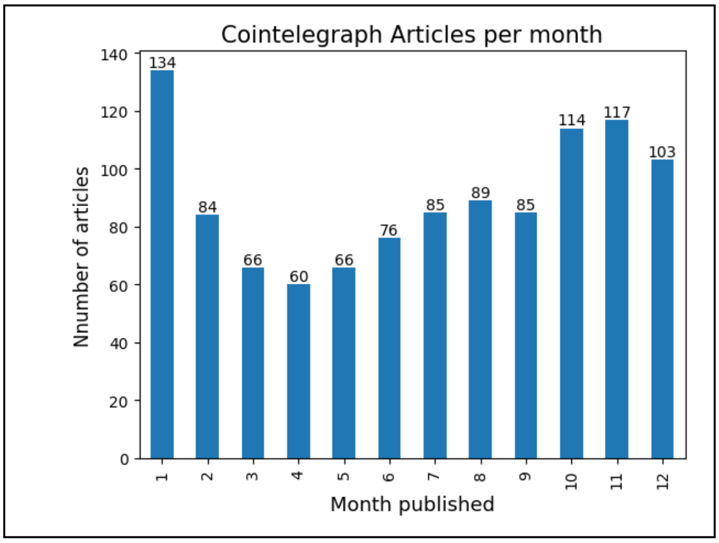
This dataset does not contain any null values or duplicated rows, but just like the previous dataset, it contains many columns which are not needed. The “Publisher” column is a strange one, as it only contains one value, that being “Cointelegraph”.

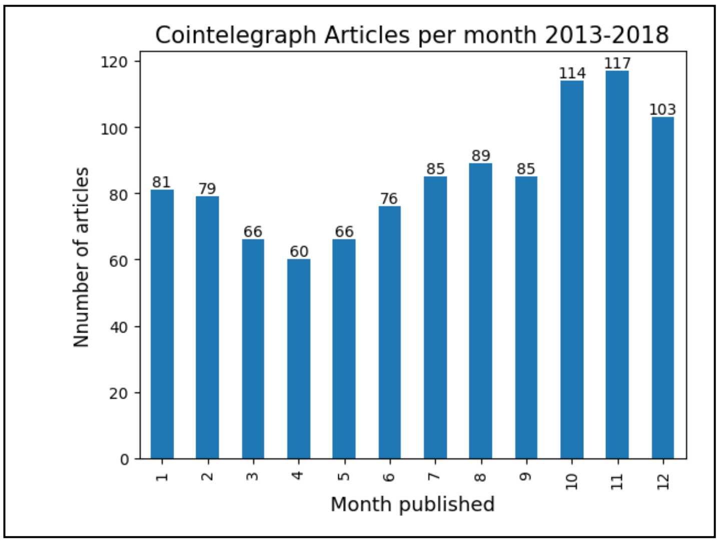
There are 113 unique authors in this dataset, thus each author wrote an average of 9.55 articles. When we look at the actual data however, seventy-seven (68.14%) of the authors only wrote one article, and twenty-four (21.23%) wrote two articles. William Suberg wrote 167 (15.48%) of the total and Marie Huillet came in second with 72 (6.67%) of the total.

For the years these articles were published during, each year has more articles published than the previous. 2019 is the only exception to this as the dataset was collected sometime during this year. Only one article here was published in 2013 while 435 (40.32%) were in 2018.

When the months each article was posted in is analysed, it is seen that they are far more equally distributed than in the BBC dataset. Here, April, the month with the least number of headlines, has sixty instances while January, the month with the most, has one hundred and thirty-four. As the headlines from 2019 stop at the beginning of February, if they are taken out it is seen that the range is decreased significantly.



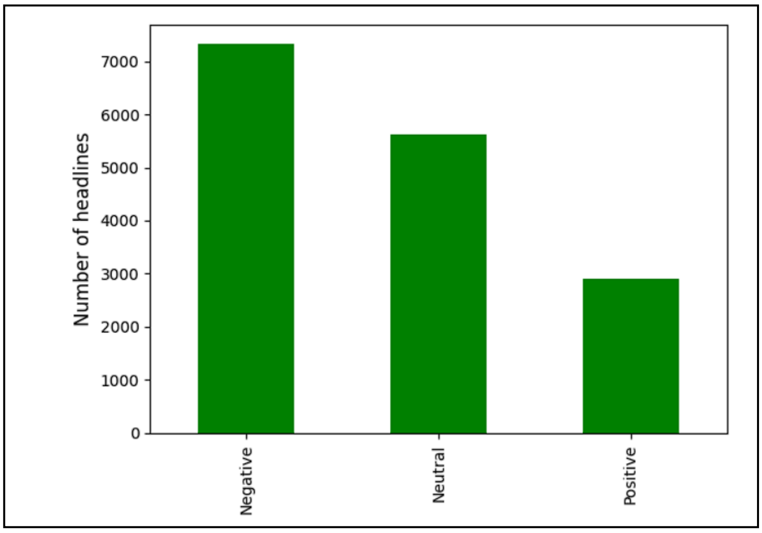




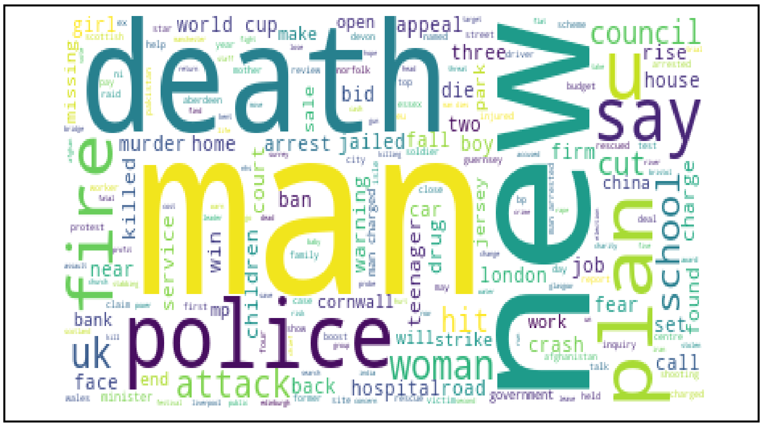
VII. Pre-Processing

A. BBC

Before making any changes to the corpus I investigated the sentiment analysis of the unchanged headlines using NTLK's SentimentIntensityAnalyzer. It claims that 46.25% of the headlines are negative, 35.45% are neutral and that only 18.3% of headlines are positive.



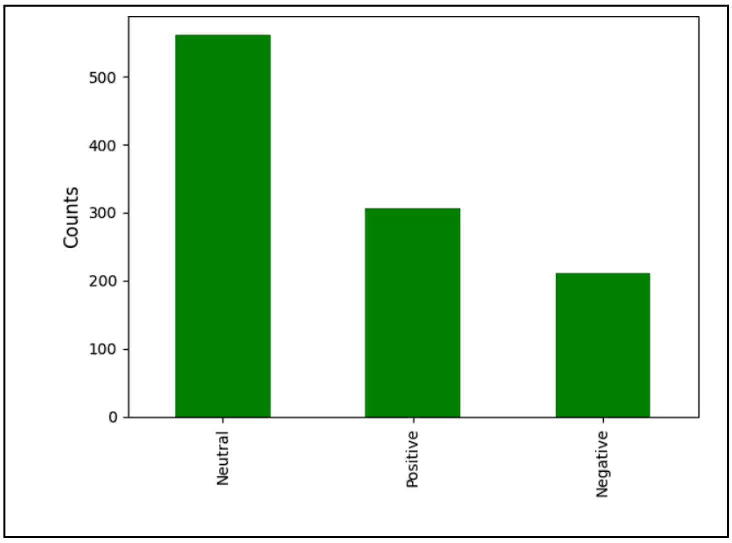
I then removed any punctuation from the headlines and made each word lowercase. Next, I removed the stop words from each headline before splitting each headline into tokens. After this I created a word cloud highlighting the words seen most frequently. This reinforces the findings from the above graph as some of the most common words include "death", "fire", "attack", "die" and "hit".



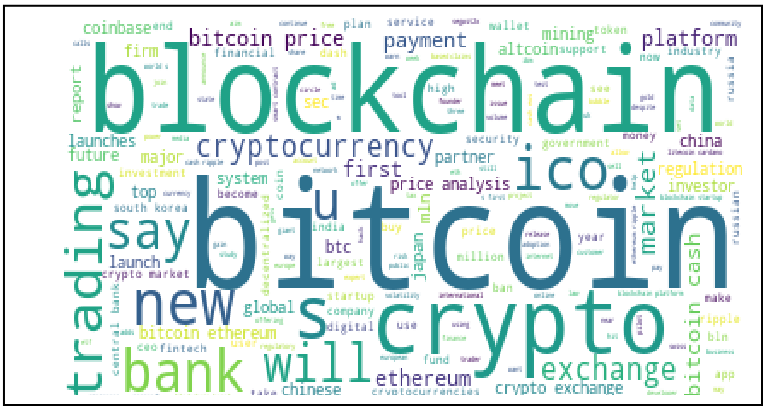
Following this I created my dictionary using Gensim's corpora.Dictionary module before creating a variable for my corpus which began as a list of each tokenized headline in its own sub list which I then used Gensim's doc2bow model on to create my bag of words. The BBC headlines were now ready for my LSA and LDA models.

B. Cointelegraph

For these headlines, I repeated the aforementioned steps. After my EDA I performed the sentiment analysis on the headlines and this time 52% of the headlines were neutral, while 28.45% were positive and the final 19.56% were negative.



Headlines had punctuation and case removed, were split and had their stop words removed before I created my dictionary and corpus, again using the doc 2 bag of words module. I created another word cloud which also reinforced the findings in the above graph as the majority of the words seen have no positive or negative connotation e.g., "blockchain", "bitcoin", "crypto" "trading". The Cointelegraph headlines were ready for the models.



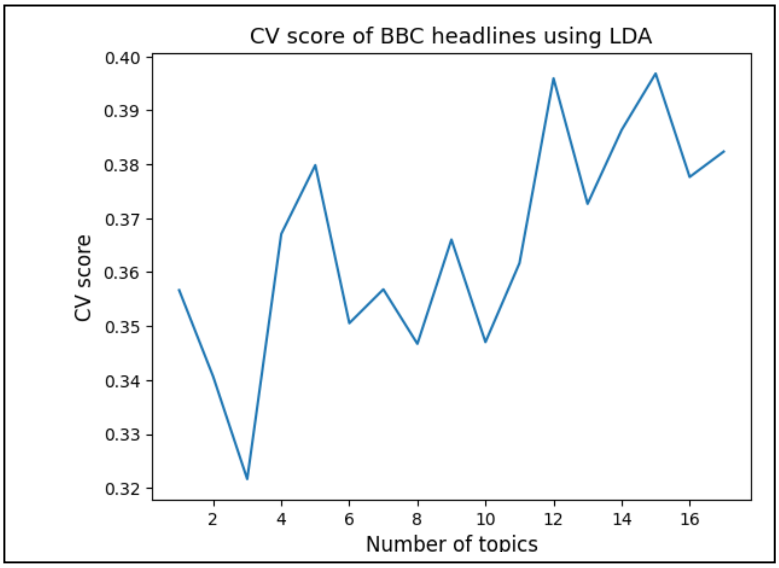
VIII. Results

For my models I used Gensim's LdaMulticore and LsiModel, as well as their CoherenceModel to calculate the coherence score of my LSI and LDA models. I used fifty iterations for each model to improve performance.

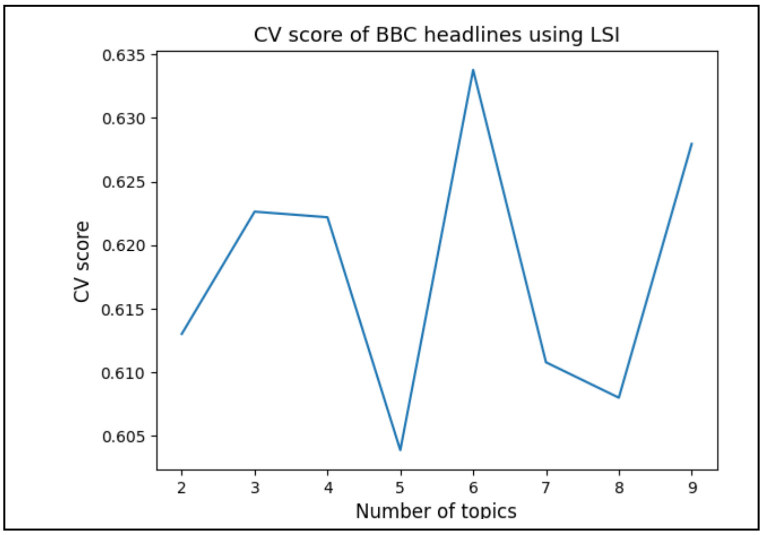
A. BBC

When looking at the coherence score returned from using LDA on the BBC headlines, we can see that as the number of topics is increased, the coherence score also increases.

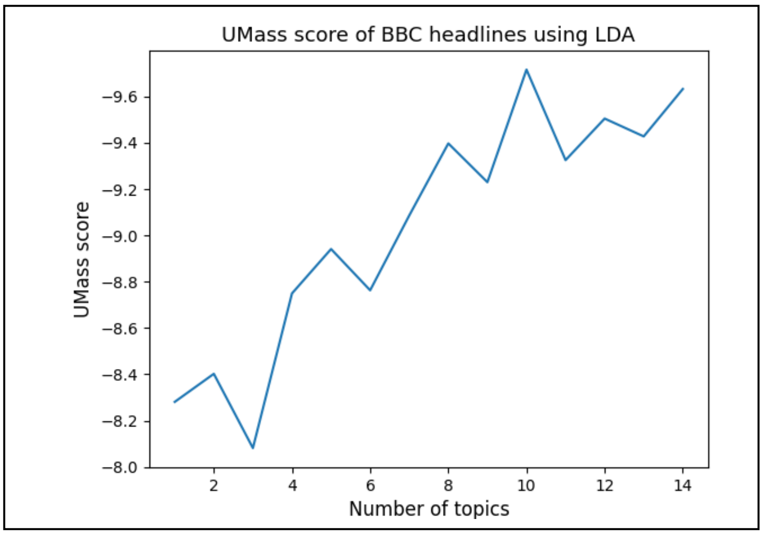
The graph tells us that five topics should be ideal as the coherence score drops off immediately after that and we don’t want to have too many topics in our model it will diminish our results.



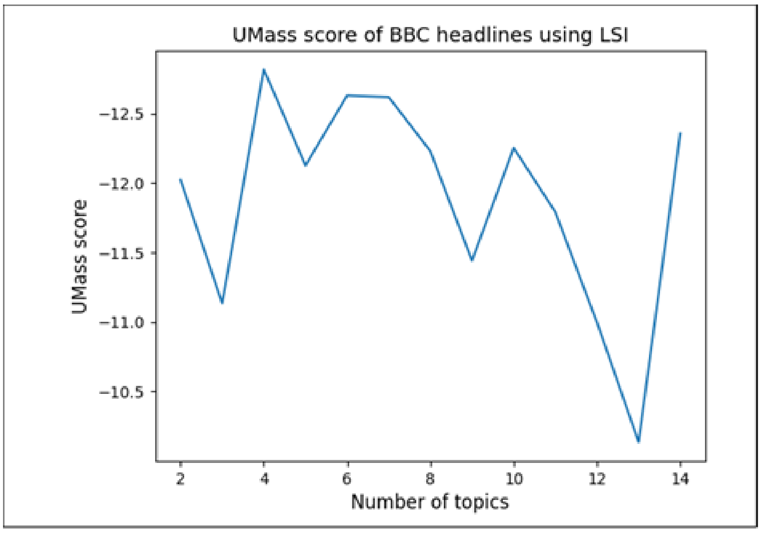
When LSI is used however, the coherence score returned is much higher than received from the LDA model. Here, six topics returns the highest coherence score and surprisingly, five topics returns the least.



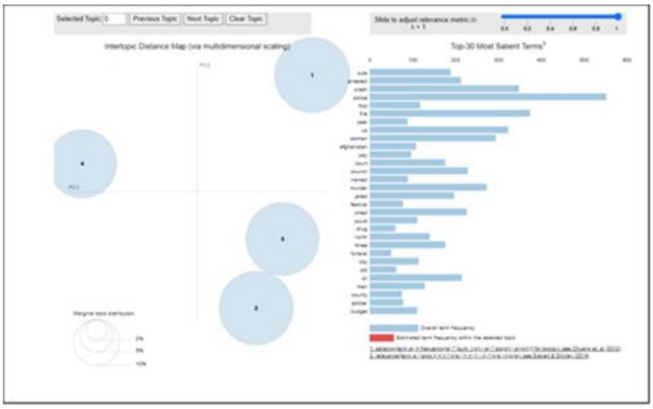
The UMass score tells us how often two words, w_i and w_j appear together in the corpus. Its value is between 14 and -14. Here we are looking to get as close to -14 as possible. When we look at the UMass score returned as the umber of topics increases, we can see that when using LDA, there is almost a direct correlation between the two metrics.



When we use the LSI model however, we see almost the opposite of LDA. Here, there is a negative correlation between the number of topics and the coherence score returned. From these results, using four topics would make our model as optimal as possible.



An interactive dashboard can be found in the Jupyter notebook visualising the results from using LDA. Unfortunately, I am unable to find a module which can be applied to the results from LSI.


Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:
Intertopic Distance Map (via multidimensional scaling)
Top-30 Most Salient Terms
Overall term frequency
Estimated term frequency within the selected topic
Marginal topic distribution

When LSI is used with six topics a TF-IDF corpus, it is clear which words are much more prominent than others in each topic. The table below contains some of the words seen in the top 10 most important for each topic. As you can see, in Topic 2, “Crash” has a strong positive value while “Man” has a significant negative value.

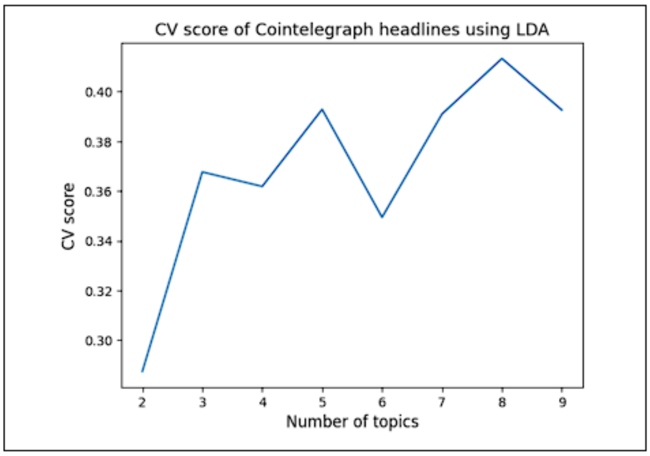
BBC Headlines after LSI with 6 Topics

Topic No.	Man	Crash	Cowell	Fire	Murder	Police	Dies
1	-0.6	-0.23	N/A	N/A	-0.2	-0.15	-0.17
2	-0.31	0.57	N/A	N/A	-0.2	N/A	0.23
3	0.3	0.18	-0.377	-0.13	N/A	-0.37	0.12
4	0.28	N/A	0.474	-0.38	N/A	-0.12	N/A
5	N/A	N/A	N/A	0.57	N/A	-0.34	0.2
6	0.31	N/A	N/A	N/A	-0.29	-0.28	0.2

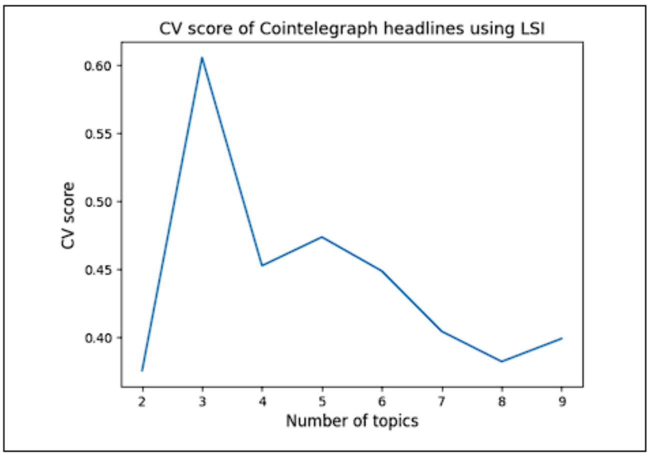
For some reason I was unable to get a coherence score using my HDP model on the BBC headlines (although as you’ll see later there were no issues getting a score for the other dataset). The model automatically chose for there to be nineteen topics. Many of these topics give the highest score to words seen in the above table like “police” having the highest score in topic 6, but it also gives high scores to words not highlighted from the other models. An example of this would be topic one, which has the word “barn” given the highest score here.

B. *Cointelegraph*

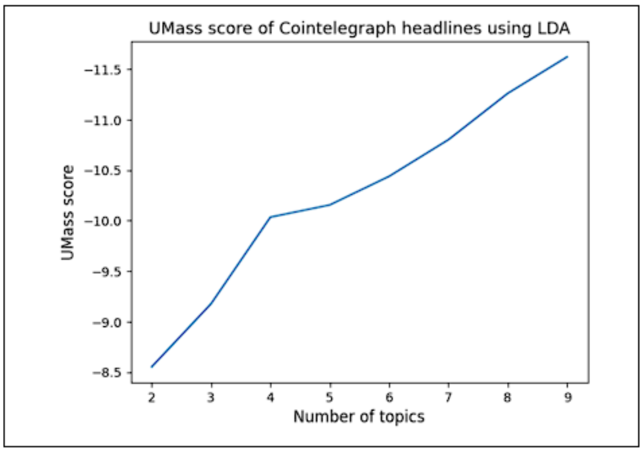
The headlines from Cointelegraph have a very similar coherence score to BBC’s when we look at their CV score using LDA. Both start very close to 0.3 but instead of needing twelve topics to get to 0.4, these headlines only need five topics to get a similar score.



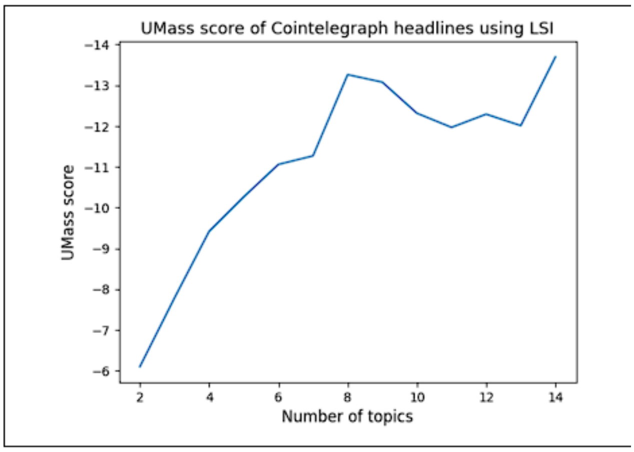
The coherence score for these headlines is much different than previously seen. Whereas the coherence score when CV was used with LSI increased and decreased within a noticeable range, here the score immediately peaks at a strong 0.61 with three topics before decreasing almost linearly to 0.38 when the number of topics is eight.



The two UMass results here each return higher scores than previously seen. Using LDA, there is almost a direct correlation between the number of topics and UMass score returned. With two topics a score of just under -8.5 is returned while at nine topics, that score goes past -11.5. These results show similar trends to what was just discussed regarding the CV score returned when run with an LDA model. Using this logic, one would expect when the UMass score is returned from an LSI model, it will behave like CV and peak at the beginning before trending downwards (here that would mean an increasingly positive coherence score).



What’s actually seen with LSI is that it behaves much like the LDA model directly above, meaning it begins with a low coherence score (-6.1) and almost always returns an increasingly lower score until it reaches -13.7 when the number of topics is set to 14.



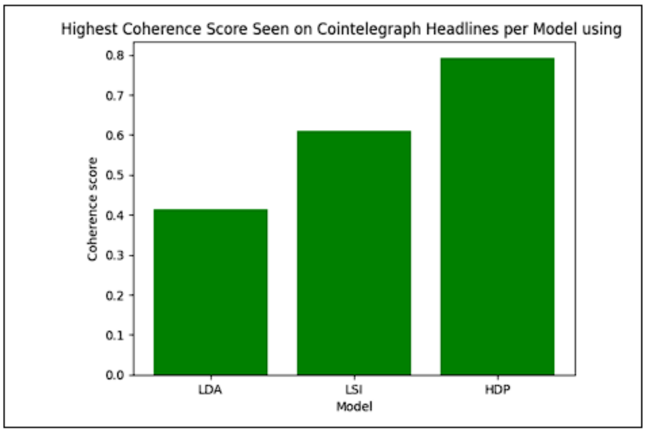
According to these results, there are multiple numbers of topics one could use to return the highest coherence score. This cannot be true in reality however, thus we must decide on the optimal number to use. According to the LDA results, using eight topics should make the model perform the best, although in reality eight topics may be too broad, there could be a lot of overlapping between them.

The LSI results are much less straightforward. Here, using three topics is clearly optimal according to the CV results however when the UMass results are taken into account, using three topics may not be enough.

The highest coherence score is seen when we run Cointelegraph's headlines through a HDP model with a BoW model used to create our term document frequency variable "corpus". Here, the HDP model returns a coherence score of 0.793. As the number of topics cannot be set manually with this type of model, the number of topics chosen by our HDP model is 19. When we look at each topic, we see that there are usually one or two groups with a higher score than the others, such as "exchange", "insider" and "integration".

IX. Conclusion

This coherence score returned by the HDP model is clearly the highest achieved throughout this project, followed by LSI, with LDA taking final place. If one were to attempt to improve these results, they could perhaps add words into the list of stop words, such as the names of places, mainly for the BBC headlines, and the names of cryptocurrency coins for Cointelegraph's. Names of people and businesses could also be removed from both, although attempting to remove all of these tokens from each dataset would prove to be incredibly tedious, given the number of instances.



References

[1] Cointelegraph *About Cointelegraph* Available at: https://cointelegraph.com/about [Accessed 10/11/2022].

[2] Cointelegraph *Advertise with* Us Available at: https://cointelegraph.com/advertise [Accessed 10/11/2022].

[3] *MachineLearningMastery A Gentle Introduction to the Bag-of-Words Model* Available at: https://machinelearningmastery.com/gentle-introduction-bag-words-model/ [Accessed 08/11/2022].

[4] Analytics Vidhya *Latent Dirichlet Allocation* Available at : https://medium.com/analytics-vidhya/latent-dirichelt-allocation-1ec8729589d4 [Accessed 10/11/2022].

[5] MonkeyLearn *Introduction to Topic Modeling* Available at: https://monkeylearn.com/blog/introduction-to-topic-modeling/ [Accessed 10/11/2022].

[6] DataCamp *Latent Semantic Analysis using Python* Available at: https://www.datacamp.com/tutorial/discovering-hidden-topics-python [Accessed 10/11/2022].

[7] Teh, Y.W.T, Jordan, M.I.J., Beal, M.J.B, Blei, D.M.B. (2005) *Hierarchical Dirichlet Processes* Available at: https://www.gatsby.ucl.ac.uk/~ywteh/research/npbayes/jasa2006.pdf [Accessed 24/11/2022].

[8] Tong, Z. Zhang, H. (2016) 'A Text Mining Research Based on LDA Topic Modelling', in *The Sixth International Conference on Computer Science, Engineering and Information Technology (CCSEIT 2016)*. Vienna, Austria. May 21-22, 2016, pp 201-210. Available at: https://www.researchgate.net/profile/Solomia-Fedushko/publication/331276764_Proceedings_of_the_Sixth_International_Conference_on_Computer_Science_Engineering_and_Information_Technology_CCSEIT_2016_Vienna_Austria_May_2122_2016/links/5c6fcd63299bf1268d1bc2b0/Proceedings-of-the-Sixth-International-Conference-on-Computer-Science-Engineering-and-Information-Technology-CCSEIT-2016-Vienna-Austria-May-2122-2016.pdf#page=212 [Accessed 11/11/2022].

[9] Y. Kalepalli, S. Tasneem, P. D. Phani Teja and S. Manne, "Effective Comparison of LDA with LSA for Topic Modelling," *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2020, pp. 1245-1250, doi: 10.1109/ICICCS48265.2020.9120888..

[10] Williams, T, Betak, J. (2018) 'A Comparison of LSA and LDA for the Analysis of Railroad Accident Text.' *Procdia Computer Science*, 130, pp.98-102