Unit Title: Natural Language Processing for the Creative Industries
Element 1: Critical Essay

Korina Zaromytidou.

Word count: 1815

This project will be focusing on the semantic analysis and result comparison using LSA and LDA topic modelling techniques on a specific dataset. The project will iterate through the outcomes of different experiments using these techniques, exploring the results and suggesting possible future alterations and developments.
The dataset used on this project has been taken from the Kaggle website (Iron486, Putin and Zelensky Tweets 2022) and is one of the 2 datasets existing under the same dataset titled 'Putin and Zelensky Tweets'. The first dataset, which is the one that will be explored, includes top daily tweets containing the keyword 'Putin'. The dataset is a 4MB CSV file, presenting tweets' data, posted from around the globe between the period 01/01/2022 to 17/07/2022, categorised under 11 columns.

Overall, topic modelling algorithmic of word interpretation and categorisation are built around the principle that the semantics of a text are regulated by 'latent' variables, which could be uncovered and shaped to provide possible meanings. Latent Semantic Analysis (LSA), introduced by Jerome Bellegarda in 2005 (Ma, 2018), was among the first methodologies that was presented during the term for the Natural Language Processing for the Creative Industries unit. This technique allows for a deeper exploration of data, compared to previously explored Bag of words (BOW) approach, in matrix format. A technique that attempts to 'leverage the context around the words to capture the hidden or latent concepts, which are called topics' (Goyal, 2021). '*Past NLP experimenters found an algorithm for revealing the meaning of word combinations and computing vectors to represent this meaning. It's called latent semantic analysis (LSA). And when you use this tool, not only can you represent the meaning of words as vectors, but you can use them to represent the meaning of entire documents.'* (Lane et al., 2019).

The dataset that aimed to be explored for this project is a collection of tweets. More specifically, the overall aim of the project is to extract the main topics, represented as a set of words, that occurred in the collection of the different tweets that were posted during the period 01/01/2022 to 17/07/2022, which occurred slightly before the Russian invasion to Ukraine on 24th February 2022. Without delving into any type of judgements on the events, how these were perceived at the time and how these appreciations developed over time, this project is seen as an opportunity to utilise the potential that such research could offer, to practice skills and possibly offer a taste of the understanding and appreciation of the global social media's community response to the topic – in this case the semantic associations to the word/ name 'Putin', referring to Vladimir Putin, the President of Russia.

Considering the nature of the explored dataset, LSA was selected as an appropriate approach for this small-scale project that could provide some interesting analytics, without looping into the temptation of prefixed topics. While exploring the potential of LSA topic modelling technique, and as a complete beginner (on both, topic modelling and coding), the project developed beyond its intended aims, and I started questioning how, or whether, a topic modelling technique of similar nature, such the Latent Dirichlet Allocation (LDA) could produce different outputs on the same dataset analysis. Latent Dirichlet Allocation (LDA), introduced by David Blei, Andrew Ng and Michael O. Jordan in 2003((Ma, 2018), is another unsupervised topic modelling, possibly a more contemporary approach, compared to LSA, that takes a probabilistic view. Considering that LSA is perceived to be the semantic analysis that *'focus on reducing the matrix dimensions, compared to the LDA that solves topic modelling problems'* (Ma, 2018), possibly the results from the application of the 2 techniques, might not be highly different, yet it is worth discovering the settle differences of the statistical co-occurrences of the words. In summary the project would be exploring in its first stage, the comparative results when applying LSA in the same data, using TruncatedSVD to compute between 5 and 15 topics vectors, and on its second phase, create and compute 15 topic vectors, using a tokenizer, for LDA analysis, comparing results with the previously computer LSA outputs.

The project was developed predominantly using the code for the LSA and LDA analysis provided during the Week 2.2 Lab (McCallum, NLP - Week 2.2. LDA/Discussion 2022) with some slight alterations. Using a copy of the jupyter notebook, I started the project by importing the relevant libraries such as Pandas, sklearn and nltk (had to import them using RIP as an addition to the code on the notebook). I uploaded the data CSV file the relevant data folder, and utilised the code to read the file into a pandas data frame. As a starting point, I printed the shape of the file, which included 10737 rows and 11 columns. I had previously checked opened the file in Excel so I could have an overall understanding of the data that was included. Using the code of the 2.2 Week lab, I changed first the name of column that was already titled 'text' and renamed the column that included the tweet at text to align with the existing code – at there was column with that name already, I had to rename also that column. Following the given code, I applied the LSA code, starting from creating count vectors using the tokenizer provided, created tf-idf vectors from your count vectors, subtracted the mean as key requirement, and then used TruncatedSVD to compute the LSA topic vectors. I used 15 topics and 30 iterations on that document (NLP Project -Putin Tweets-B - 15 topics). As I wanted to explore the possible outcomes with a different number of topics, I created a copy of that file saving it as NLP Project -Putin Tweets-B – 5 topics, but this time I initiated 5 topics, saving the results.

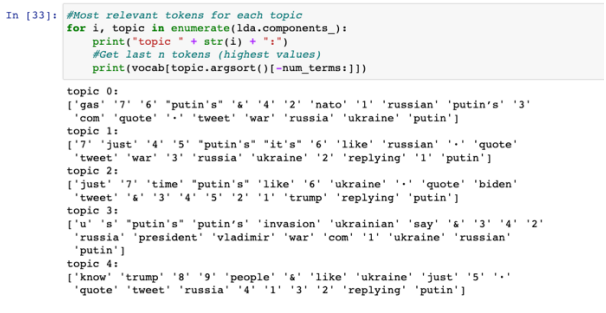
For both files I applied the Latent Dirichlet Allocation, using the code from the lecture notebook – session 2.2. Again, I computed the count vectors and used the tokenizer that was provided to split the text into sentences and the sentences into words, yet this time it was a bag of words calculated rather than a TFIDF.

It is worth mentioning at the beginning of the project, I was planning on exploring the comparison of the LSA results between 5 and 10 topics (on same data). However, when looking at their outputs, there didn't seem to be any significantly different results – for example when computing the highest weighted words for each topic, the results for the first 5 topics were practically the same as of the 10topics, while the words seem to be highly repeated for all topics. I assume that this was representative of the frequency that each word was used, however it did not allow for any semantically meaningful comparisons. Reflective on this observation, I decided to change the number of the topics, exploring a comparison between 5 and 15 topics, investigating whether a more intense summation, (heightening the numeric difference of the topic explored) might incur more obvious variety. Exploring the results, it was evident that although the weighted relevance between tokens, tweets and topics was very low in both cases, when printing the highest weighted words for each topic (for this case of 15 topics), the result provided more varying results. The 15 topic results were presenting some new words, that were not present at the equivalent 5 topics results *(figure 01, figure 02).* Words like 'European', 'energy', 'sanction', 'crime' and 'people' were part of the highest valued, providing more meaningful / or possible appreciation of the topics that these were weighted against. Similarly, as on a more sociological approach, one would possibly anticipate finding these words on relevant comments.

```
In [66]: num_terms = 20
         for i in range(num_topics):
             print("___topic " + str(i) + "___")
             topicName = "topic" + str(i)
             weightedlist = topic_weights.get(topicName).sort_values()[-num_terms:]
             print(weightedlist.index.values)

___topic 0___
['-' 'quote' 'just' 'vladimir' 'tweet' 'president' '&' '6' 'trump' '5' '4'
 '3' 'war' 'russian' 'russia' '2' 'ukraine' 'replying' '1' 'putin']
___topic 1___
['people' "it's" 'thing' 'right' "don't" 'republican' '5' '3' '14' '6'
 'did' 'think' 'just' 'know' 'like' '2' 'putin' 'trump' '1' 'replying']
___topic 2___
['people' 'like' '3' 'biden' 'republican' 'gas' 'twitter' 'price' 'feb'
 'mar' 'apr' 'jun' 'jan' 'jul' '&' 'thread' 'trump' '-' 'quote' 'tweet']
___topic 3___
['white' 'gop' 'russian' 'leader' 'house' 'meeting' 'election' 'joe' 'say'
 'republican' 'said' 'donald' 'com' 'u' 's' 'putin' 'vladimir' 'biden'
 'president' 'trump']
___topic 4___
['com' 'inflation' '9' 'price' 'crime' 'ukrainian' '6' 'putin's' '5'
 'quote' '-' 'biden' 'tweet' 'trump' 'war' 'vladimir' 'president' '4'
 'russian' '1']
___topic 5___
['@potus' 'europe' 'supply' 'company' '2' 'american' 'it's' "putin's"
 'putin's' 'joe' 'blame' 'hike' 'energy' 'high' 'inflation' 'war' 'oil'
 'biden' 'gas' 'price']
___topic 6___
['donald' 'criminal' 'view' '4' 'people' 'following' 'potentially'
 'setting' 'content' 'sensitive' 'change' 'includes' 'medium' "putin's"
 'crime' 'putin's' '3' '&' 'war' 'trump']
___topic 7___
['putin's' 'oil' "putin's" 'price' 'gas' 'setting' 'potentially'
 'includes' 'sensitive' 'following' 'content' 'view' 'change' 'president'
 'people' 'ukrainian' 'medium' 'vladimir' '2' 'russian']
___topic 8___
['need' '4' 'ally' 'country' 's' 'u' '1' 'biden' '&' 'nato' 'russia'
 'following' 'view' 'content' 'setting' 'potentially' 'sensitive' 'change'
 'includes' 'medium']
___topic 9___
['sensitive' 'view' 'want' 'stop' 'following' 'content' 'includes'
 'setting' 'potentially' 'replying' "don't" 'vladimir' 'just' 'think' '6'
 'know' 'like' 'ukraine' 'putin' '3']
___topic 10___
["it's" 'energy' 'europe' "putin's" 'ukrainian' 'nato' 's' 'price' 'u'
 'replying' 'country' 'sanction' 'oil' 'people' 'gas' '4' 'russia'
 'russian' '3' '&']
___topic 11___
['don't' 'power' 'look' 'doe' 'make' 'know' 'it's' 'crime' 'vladimir'
 'just' 'russia' 'world' 'president' 'people' 'biden' 'putin' 'want' 'war'
 '4' 'like']
___topic 12___
['need' 'gas' 'want' 'putin' 'like' 'invading' '&' 'invaded' 'stop'
 'price' 'invade' 'think' 'just' 'ukrainian' '10' 'invasion' 'biden'
 'people' '4' 'ukraine']
___topic 13___
```

Fig 01(Zaromytidou , *NLP Mini Project -Putin Tweets-B -15 topics*)



```
In [33]: #Most relevant tokens for each topic
         for i, topic in enumerate(lda.components_):
             print("topic " + str(i) + ":")
             #Get last n tokens (highest values)
             print(vocab[topic.argsort()[-num_terms:]])

topic 0:
['gas' '7' '6' "putin's" '&' '4' '2' 'nato' '1' 'russian' 'putin's' '3'
 'com' 'quote' '-' 'tweet' 'war' 'russia' 'ukraine' 'putin']
topic 1:
['7' 'just' '4' '5' "putin's" "it's" '6' 'like' 'russian' '-' 'quote'
 'tweet' 'war' '3' 'russia' 'ukraine' '2' 'replying' '1' 'putin']
topic 2:
['just' '7' 'time' "putin's" 'like' '6' 'ukraine' '-' 'quote' 'biden'
 'tweet' '&' '3' '4' '5' '2' '1' 'trump' 'replying' 'putin']
topic 3:
['u' 's' "putin's" 'putin's' 'invasion' 'ukrainian' 'say' '&' '3' '4' '2'
 'russia' 'president' 'vladimir' 'war' 'com' '1' 'ukraine' 'russian'
 'putin']
topic 4:
['know' 'trump' '8' '9' 'people' '&' 'like' 'ukraine' 'just' '5' '-'
 'quote' 'tweet' 'russia' '4' '1' '3' '2' 'replying' 'putin']
```

Fig 02 (Zaromytidou ,*NLP Mini Project -Putin Tweets-A-5 topics)*

The comparison between LSA and LDA techniques it was monitored for both cases (5 topics and 15 topics). On both occasions what was interesting was that in the question on how much each topic apply to each tweet, when using the LDA method, it was evident that at least for one topic the values was much higher, creating a spike, proving that these were more closely aligned to one topic. While with LSA, there seems to be much bigger spread, that doesn't prove a strong alignment with any of the topics. This was the case when comparing LSA and LDA in both scenarios of the 5 or 15 topics retrospectively. When printing the highest achieved words, the 15-topic LSA method showed, as discussed above, a bigger variety of words, while the 5-topic output was quite restricted to repeated words. What was interesting is that the print-out of the highest achieved words for LDA (15 words) were also quite repetitive *(figure 03),* with a quite restricted vocabulary compare to the LSA output.

```
In [76]: #Most relevant tokens for each topic
         for i, topic in enumerate(lda.components_):
             print("topic " + str(i) + ":")
             #Get last n tokens (highest values)
             print(vocab[topic.argsort()[-num_terms:]])
```

```
topic 0:
['#putin' '6' '&' '#ukraine' 'russian' '4' "putin's" '7' 'com' '2' '3' '·'
 'quote' 'war' 'tweet' '1' 'russia' 'nato' 'ukraine' 'putin']
topic 1:
['5' '6' '4' '·' "don't" 'quote' 'tweet' 'just' 'russian' 'war' "putin's"
 "it's" 'like' '3' 'russia' 'ukraine' '2' 'replying' '1' 'putin']
topic 2:
['democracy' '11' 'just' '&' 'time' '·' 'quote' 'tweet' 'like' 'ukraine'
 '3' "putin's" 'biden' 'trump' '4' '5' '2' '1' 'replying' 'putin']
topic 3:
["putin's" 'military' '&' 'biden' 'invasion' '3' 'say' '2' '4' 'war' 's'
 'u' 'russia' '1' 'com' 'vladimir' 'president' 'ukraine' 'russian' 'putin']
topic 4:
['country' '9' 'think' 'war' 'know' "putin's" '&' '5' 'just' 'like' '3'
 '·' 'quote' 'tweet' '4' 'russia' 'replying' '2' '1' 'putin']
topic 5:
['6' '4' '10' 'russia' 'trump' 'people' '5' '&' 'war' 'just' 'russian'
 'ukraine' '3' 'quote' '·' 'tweet' '2' '1' 'replying' 'putin']
topic 6:
['5' 'vladimir' 'invasion' 'com' '4' "putin's" 'fuel' 'putin's' 'gas' '&'
 'price' '1' 'trump' 'replying' 'war' '3' 'oil' '2' 'ukraine' 'putin']
topic 7:
['putin's' '9' '&' 'like' '7' 'biden' '8' '6' 'russian' 'russia' '5'
 'quote' '·' 'ukraine' 'tweet' '3' '1' '2' 'replying' 'putin']
topic 8:
['zelensky' 'people' 'invasion' 'ukrainian' "putin's" 'said' '3'
 'vladimir' '7' '5' 'president' 'putin's' 'replying' 'war' 'com' 'russian'
 '1' 'ukraine' 'russia' 'putin']
topic 9:
['president' '4' 'putin's' '5' '&' 'com' '·' '3' 'quote' 'russia' '6'
 'tweet' 'replying' 'russian' 'biden' '2' '1' 'war' 'ukraine' 'putin']
topic 10:
['7' 'war' 'germany' 'russian' 'trump' "putin's" 'nato' '5' '·' 'quote'
 'tweet' '2' '1' '&' 'russia' '4' '3' 'ukraine' 'replying' 'putin']
topic 11:
['6' '·' 'tweet' 'quote' 'com' 'putin's' '4' 'vladimir' 'world'
 'ukrainian' '3' 'trump' 'russian' 'russia' 'war' '1' '2' 'replying'
 'ukraine' 'putin']
topic 12:
['want' 'com' 'know' 'country' '5' 'vladimir' '&' '6' 'war' 'people'
 'russian' '3' 'russia' '4' '2' '1' 'ukraine' 'trump' 'replying' 'putin']
topic 13:
['9' '&' 'tweet' '"' '·' 'quote' 'say' '3' 'flynn' 'com' '4' 'russia'
 "putin's" '5' '2' 'putin's' 'russian' '1' 'replying' 'putin']
topic 14:
['8' 'just' '·' 'quote' 'tweet' 'putin's' '5' 'com' '&' '4' 'people'
 'ukrainian' '3' '2' 'replying' 'war' 'russian' '1' 'ukraine' 'putin']
```

Fig 03 LDA Analysis Zaromytidou , *NLP Mini Project -Putin Tweets-B -15 topics*)

This was a small-scale study, that was compromised by time pressure and coding skills restrictions. Therefore, possibly it would not be fair to step into any definitive conclusions for these comparisons. Possibly a starting point would be to think that LSA might be a more effective text processing method for extracting meaningful similarities, however possibly this

might need to be explored through a variety of data, and through a more realistic comparison of different scale to achieve a more concrete conclusion. This project relied highly on the application of specific parameters as set by the existing code, and I acted mainly as the observer of its outputs. If I had more time to explore this project and the different coding avenues, I think that one of the changes I would apply would be to clean further the texts, for example by removing the numbers that seemed to be part of the tweets, possibly even the hashtags or emojis, so this would allow for more meaningful results. Possibly a study would be to explore the relationship between semantic (words) and semiotics (emojis) under the prism of topic modelling. I would also try to explore and investigate whether there is an alignment between specific names and words. I think that also a semantic analysis would be quite interesting in this set of data.

The data for this file, as download by Kagger, were combined by Web scraping. Kagger is asking its users to use the API, providing appropriate documentation, however there is no clear mention whether this was applicable in this instance. There is no mention on the dataset provided on whether a formal Data Collection Policy has been developed. The data provide by the user Iron486, who identifies as a master's student at University of Pisa, has indicated that the obtained data were 'pre-processed, filling the missing values in the columns with numerical data' ((Iron486: Expert 2022), however no further details have been provided to demonstrate ethic scrapping of information, or re-use.

References and Bibliography.

McCallum, L. (2022) "NLP - Week 2.2. LDA/Discussion," 15 December. Available at: https://ual.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=aa7dbb94-088e-447f-a9d0-af2b00b803ec.

McCallum, L. (2022) "NLP - Week 2.2. LSA," 15 December. Available at: https://ual.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=37d6fde6-8d6f-44ba-b55f-af2b00c0e309

Fiebrink, R. (2022) "NLP Week 7.1 - NLP Projects start to finish." London : UAL,Camberwell, 15 December.

Goyal, C. (2021) *Topic modelling using LSA: Guide to master NLP (part 16)*, *Analytics Vidhya*. Available at: https://www.analyticsvidhya.com/blog/2021/06/part-16-step-by-step-guide-to-master-nlp-topic-modelling-using-lsa/ (Accessed: December 15, 2022).

Iron486 (2022) *Putin and Zelensky Tweets*, *Kaggle*. Available at: https://www.kaggle.com/datasets/die9origephit/putin-and-zelensky-tweets (Accessed: December 15, 2022).

Iron486: Expert (no date) *Kaggle*. Available at: https://www.kaggle.com/die9origephit (Accessed: December 17, 2022).

Lane, H. *et al.* (2019) “Natural Language Processing in Action : Understanding, Analyzing, and Generating Text with Python,” in *Natural language processing in action understanding, analyzing, and generating text with python*. Shelter Island, NY: Manning Publications Company.

Ma, E. (2018) *2 latent methods for dimension reduction and topic modeling*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/2-latent-methods-for-dimension-reduction-and-topic-modeling-20ff6d7d547 (Accessed: December 15, 2022).

McCallum, L. (2022) “NLP - Week 2.2. LSA,” 15 December.

Mitchell, R. (2018) *Web scraping with python: Collecting more data from the modern web*. Beijing, China: O'Reilly.

Snyder , R.m (2015) “An Introduction to Topic Modeling as an Unsupervised Machine Learning Way to Organize Text Information .” North Myrtle Beach, South Carolina. Available at: https://files.eric.ed.gov/fulltext/ED571275.pdf.

Zaromytidou , K. (no date) *NLP Mini Project -Putin Tweets-B - 15 topics* .

Zaromytidou , K. (no date) *NLP Mini Project -Putin Tweets-A - 5 topics* .