HD PROJECT

COS80023- BIG DATA

PROJECT TOPIC: TOPIC MODELLING

Sheba Ann Roy

103163977

EXECUTIVE SUMMARY

This report summarizes the final project for the big data unit. In the first section, the topic selected for the project is introduced, which is topic modelling for tweets based in Australia. Then, the main steps involved in the first phase of the project, which is data extraction is described. Later, the report covers how the two selected topic models LDA and BERTopic were used to successfully extract topics from the tweets, and finally a comparison is made between the performance of the two models.

TABLE OF CONTENTS

Contents

INTRODUCTION

This project focuses on extracting tweets related to Australia from the social media platform Twitter using topic modelling, which is an approach to extract the hidden subjects from huge amounts of text to identify what topics are being talked about in these tweets. The data was first extracted using SNScrape, after which the data was preprocessed. After cleaning up the data and applying other necessary pre-processing steps, the model was applied. Here, two different models were considered, namely LDA and BERTopic.

This project covers two important learning outcomes for this unit, which are:

- Compare and apply technologies and algorithms to extract and integrate information from data sets with varying degrees of structure.
- Appraise the quality and fitness for purpose of the information extracted, using suitable statistical methods.

DATA EXTRACTION

For data extraction, I focused on getting data from Twitter. Initially, the twitter API was considered, for which a developers account on twitter had to be created, which would provide keys that could be used to scrape tweets. However, it was realized that there was a restriction in using this API, which was the number of tweets that could be scrapped per day, which was around 5000. Therefore, I decided to use a scraper called SNScrape to scrape tweets from the platform without any restrictions, and 50,000 tweets were scraped. To filter out the search, the 'advanced search' feature of twitter was used, which was helpful in filtering data according to what we are looking for. This was used to filter only english tweets in the last five years. Using this advanced search will generate a query, which could be used with snscrape to scrape the data.

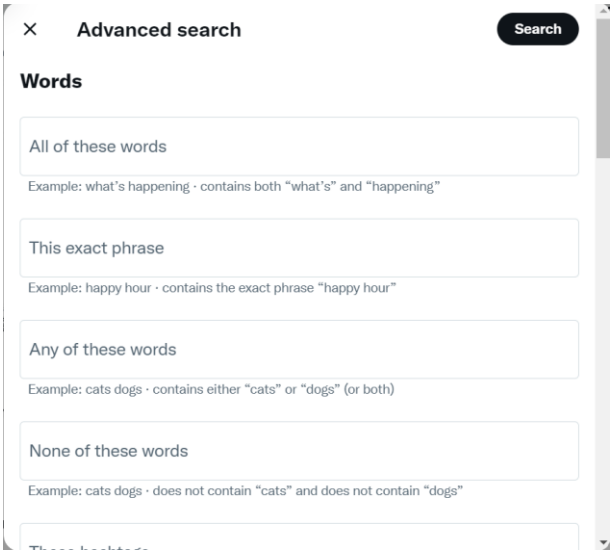

Figure 1: Advanced search feature

DATA PREPROCESSING

Once the data was extracted, it had to be preprocessed. The data was loaded into a dataframe, and the 'Tweet' column was used extract the topics.

User	Tweet	Location
CHVSEN_CVSH	There is 14 million kangaroos in Australia and 3.5 million people in Uruguay. Let's just say the kangaroos would invade Uruguay, each person would have to fight 14 kangaroos 😳	In ya moms mouth, NY
g1nfresh	The world can't heal until Ollie Pope drops 800 runs on Australia in Australia	NaN
MyFirstCousin	Rupert Murdoch, his son, his news rags, Sky News and his IPA are avoidable burdens that are costing Australia $millions. Why is this?	NaN
mysportscores	New South Wales 4 * v South Australia #CricketGame	Canada
.andDownUndead	Get out of Newcastle while you can still run. #Australia #zombieapocalypse	Australia
...	...	...
HalcyonW	Nice to see the Progress Pride flag at #PhillipIsland Australia race @MotoGP.	Rhode Island, USA
testPatrickW1	New #flood detected in Australia on 2022-10-16T19:06:45\nThis is a Twitter Test Message 6\nProvided by @CopernicusEMS #GloFAS #FloodMonitoring #GFM	NaN
BywaterFelicity	Next up, because it's spring in Australia, I need to make a hat. But I have plenty of #yarn this time.\n\nIt will be my first hat so there may be some more swearing involved.	Australia
Mintwaveradio	#nowplaying on Mintwave Radio Gramophonedzie - Why Don't You #alexa #streema #Scotland #echo #global #uk #instagram #tiktok #twitter #facebook #linkedin #podcasts #radio #usa #Finland #germany #Belgium #australia #newzealand #southafrica #canada	Global

Figure 2: Raw dataset

Most of the pre-processing was done using the Natural Language Toolkit (NLTK), which is a python library used for natural language processing. The pre-processing steps done were:

Expanding contractions

Initially all the contractions were removed from the tweets using the contractions library in python. For e.g., words like 'don't' would be converted to 'do not'. This was done so that when

punctuations are removed these words don't lose their meaning, and to reduce dimensionality, as don't' and do not would be considered different words.

Removing URL

Since most of the tweets contains links, these URLs were removed. If it is not removed, after tokenization, words like 'https' would be appearing multiple times and would affect the topic. This was done by setting a regular expression using the re library.

Removing punctuation marks

The tweets contained many punctuation marks, which wouldn't really contribute to forming a topic, so any character that was not a letter was removed. This included removing digits, emojis and any special characters as well.

Converting to lower case

The text was converted to lower case to standardize the data, using String lower () method.

Removing stop words

Stop words such as a, the, etc. are frequently used in texts. Since they don't add much value to the meaning in a document, they are removed using the list of stop words in the nltk library.

Tokenization

Tokenization is a technique used in natural language processing to break down phrases and paragraphs into simpler language-assignable elements. This was done using the word_tokenize function in the NLTK library.

Lemmatization

After tokenization, the words are lemmatized using NLTK WordNet lemmatizer to reduce it to its root form or 'lemma'. Wordnet is a publicly available lexical database present in the NLTK library. Initially, stemming was done, but this resulted in a lot of the words losing its original meaning. Therefore, lemmatization was done.

Removing single characters

However, after lemmatization, some words like 'vs' was reduced to just 'v', and this was effecting the keywords of the topics formed. To avoid this, single characters were removed.

Tweet	Location	remove_url	removed_contract	removed_punctuations	stopword_removed	tokenized_sents	lemmatized	final_col
There is 14 million kangaroos in Australia and 3.5 million people in Uruguay. Let's just say the kangaroos would invade Uruguay, each person would have to fight 14 kangaroos 😳	In ya moms mouth, NY	There is 14 million kangaroos in Australia and 3.5 million people in Uruguay. Let's just say the kangaroos would invade Uruguay, each person would have to fight 14 kangaroos 😳	There is 14 million kangaroos in Australia and 3.5 million people in Uruguay. Let us just say the kangaroos would invade Uruguay, each person would have to fight 14 kangaroos 😳	there is million kangaroos in australia and million people in uruguay let us just say the kangaroos would invade uruguay each person would have to fight kangaroos	million kangaroos australia million people uruguay let us say kangaroos would invade uruguay person would fight kangaroos	[million, kangaroos, australia, million, people, uruguay, let, us, say, kangaroos, would, invade, uruguay, person, would, fight, kangaroos]	[million, kangaroo, australia, million, people, uruguay, let, u, say, kangaroo, would, invade, uruguay, person, would, fight, kangaroo]	[million, kangaroo, australia, million, people, uruguay, let, say, kangaroo, would, invade, uruguay, person, would, fight, kangaroo]
The world can't heal until Ollie Pope drops 800 runs on Australia in Australia	NaN	The world can't heal until Ollie Pope drops 800 runs on Australia in Australia	The world cannot heal until Ollie Pope drops 800 runs on Australia in Australia	the world cannot heal until ollie pope drops runs on australia in australia	world cannot heal ollie pope drops runs australia australia	[world, can, not, heal, ollie, pope, drops, runs, australia, australia]	[world, can, not, heal, ollie, pope, drop, run, australia, australia]	[world, can, not, heal, ollie, pope, drop, run, australia, australia]
Rupert Murdoch, his son, his news rags, Sky News and his IPA are avoidable burdens that are costing Australia $millions. Why is this?	NaN	Rupert Murdoch, his son, his news rags, Sky News and his IPA are avoidable burdens that are costing Australia $millions. Why is this?	Rupert Murdoch, his son, his news rags, Sky News and his IPA are avoidable burdens that are costing Australia $millions. Why is this?	rupert murdoch his son his news rags sky news and his ipa are avoidable burdens that are costing australia millions why is this	rupert murdoch son news rags sky news ipa avoidable burdens costing australia millions	[rupert, murdoch, son, news, rags, sky, news, ipa, avoidable, burdens, costing, australia, millions]	[rupert, murdoch, son, news, rag, sky, news, ipa, avoidable, burden, costing, australia, million]	[rupert, murdoch, son, news, rag, sky, news, ipa, avoidable, burden, costing, australia, million]

Figure 3: Data after pre-processing

MODEL 1: LDA

Latent Dirichlet Allocation (LDA) is an unsupervised clustering method often used for text analysis. It treats each document as a combination of topics, and each topic is treated as a combination of keywords.

The algorithm rearranges the distribution of topics within documents and the distribution of keywords within topics to create a decent composition of topic-keyword distribution.

Simply said, a topic is a group of popular keywords and the topic's main points may be determined simply by looking at the keywords.

To implement this model, Gensim, which is an open-source python library was used as this provides an inbuilt version of LDA model.

Creating bigrams and trigrams

Bigrams are two words, and trigrams are three words occurring together in the document. These are created using gensim's phrases model.

```
 'south_africa',
 'victory',
 'australia',
 'led',
 'india_new_zealand',
 'semi_final',
 'wc',
 'south_africa',
 'victory',
 'india',
 'might',
 'led',
 'india_new_zealand',
 'semi_final',
 'hopefully',
 'different',
 'result',
 'time',
```

Figure 4: Bigrams and trigrams

Building Dictionary & Corpus for Topic Model

For the model, each tweet is considered as a document, and a corpus is a collection of these documents. 'gensim.corpora.Dictionary' class is used to associate each word in the corpus with a unique integer ID, thus forming a dictionary. This dictionary is then used to create a bag of words, as Gensim requires corpus input in the form of a bag of words or a tf-idf dictionary to create the LDA model.

```
[[(0, 1),
  (1, 1),
  (2, 1),
  (3, 1),
  (4, 1),
  (5, 1),
  (6, 1),
  (7, 1),
  (8, 1),
  (9, 1),
  (10, 1),
  (11, 2)],
 [(0, 2), (12, 1), (13, 1), (14, 1), (15, 1), (16, 1), (17, 1), (18, 1)],
 [(0, 1),
  (5, 1),
  (19, 1),
  (20, 1),
  (21, 1),
  (22, 1),
  (23, 1),
  (24, 1),
  (25, 1),
  (26, 1),
  (27, 1)],
 [(0, 1), (28, 1), (29, 1), (30, 1)],
 [(0, 1), (17, 1), (31, 1), (32, 1), (33, 1), (34, 1)],
```

Figure 5: Dictionary created

```
[[('australia', 1),
  ('fight_kangaroo', 1),
  ('invade_uruguay', 1),
  ('kangaroo', 1),
  ('let', 1),
  ('million', 1),
  ('million_kangaroo', 1),
  ('people', 1),
  ('person', 1),
  ('say', 1),
  ('uruguay', 1),
  ('would', 2)]]
```

Figure 6: Human readable form

Hyper-parameter tuning

To find the optimal value for number of topics, hyper-parameter tuning is done for 'num_topics', which is tuned for values 1 to 10, using a simple for loop and the final number of topics is selected based on the highest coherence score.

```
p=[]
c={}

for i in range(1,11):
    lda_model = gensim.models.ldamodel.LdaModel(corpus=corpus,
                                            id2word=id2word,
                                            num_topics=i,
                                            random_state=50,
                                            update_every=2,
                                            chunksize=500,
                                            passes=5,
                                            alpha='auto',
                                            per_word_topics=False)
    #Compute Perplexity
    print('\nPerplexity for ',i,": ", lda_model.log_perplexity(corpus))  # a measure of how good the model is. lower the better.
    p.append(lda_model.log_perplexity(corpus))

    # Compute Coherence Score
    coherence_model_lda = CoherenceModel(model=lda_model, texts=data_words_trigrams, dictionary=id2word, coherence='c_v')
    coherence_lda = coherence_model_lda.get_coherence()
    print('\nCoherence Score for ',i,": ", coherence_lda)
    #c['i'].append(coherence_lda)
    #c["score"].append(coherence_lda)
    c.update({i: coherence_lda})
```

Figure 7: Tuning for number of topics

Output

After hyperparameter tuning, the highest coherence score of 0.49 was obtained when number of topics are set to 9.

```
Perplexity:  -8.919694944194623

Coherence Score:  0.49045178191952604
```

Coherence score in topic modeling is used to measure how interpretable the topics are to humans. The level of perplexity indicates how effectively a model predicts a sample, and it should decrease as number of topics increases.

Visualizing results

After the LDA model had been constructed, the extracted topics and related keywords are examined using pyLDAvis package's interactive chart.

Each bubble denotes a certain topic. The greater the size of the bubble, the more tweets in the corpus that are related to that topic. The total frequency of each keyword in the corpus is shown by blue bars. Red bars indicate number of times a given keyword was generated by a given topic. As we can see from the image below, there are about 60,000 of the keyword 'Australia, and this is used about 30,000 times within topic 1.

The distance between the bubbles determines how distinct the topics are from one another.

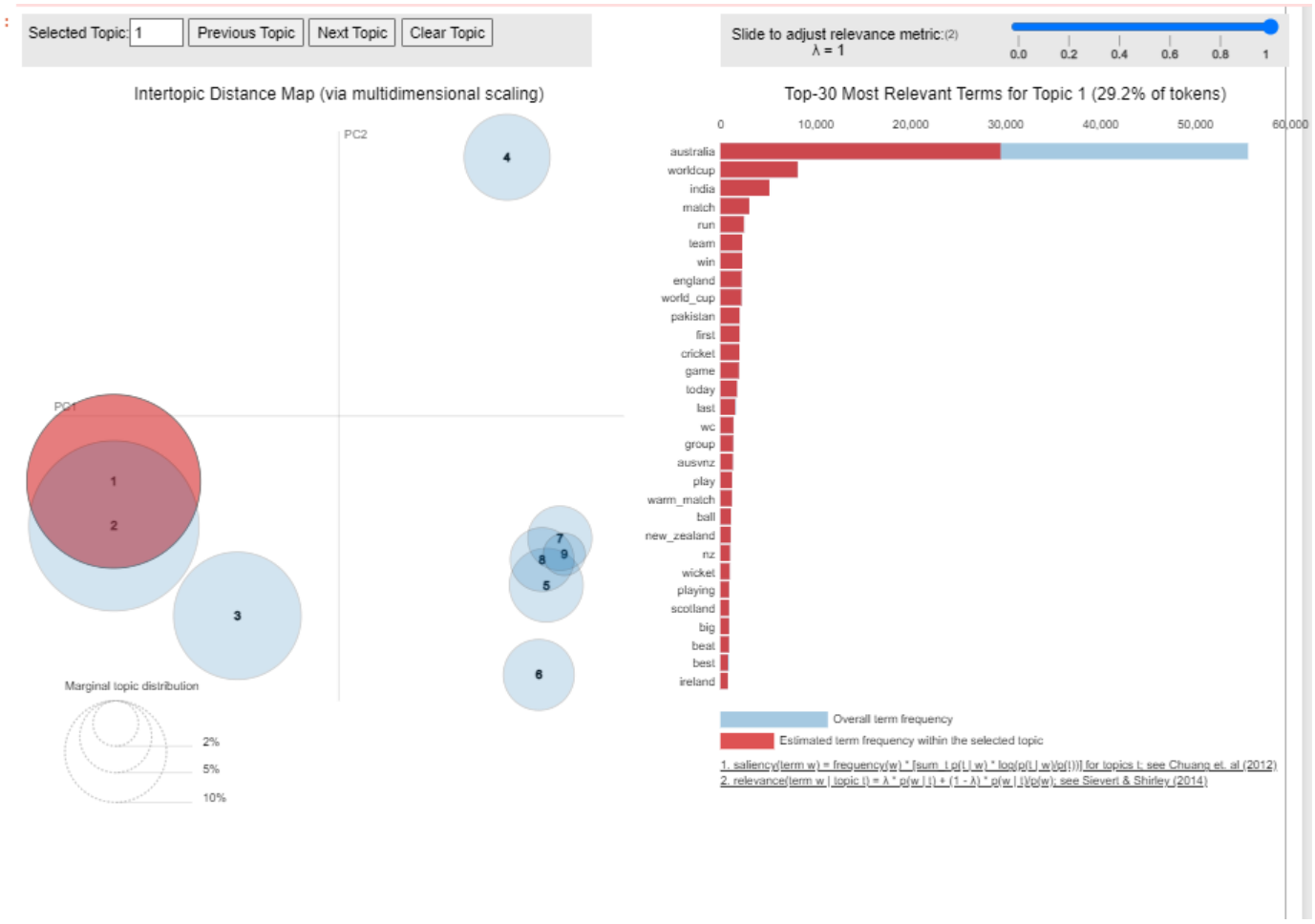

Figure 8: pyldavis visualization

We can see that topics 1, 2 and 5, 7, 8, 9 are overlapping with each other, meaning that these topics are similar.

A word cloud was also generated for the topics generated, which is helpful in identifying dominant keywords in each topic with a quick glance

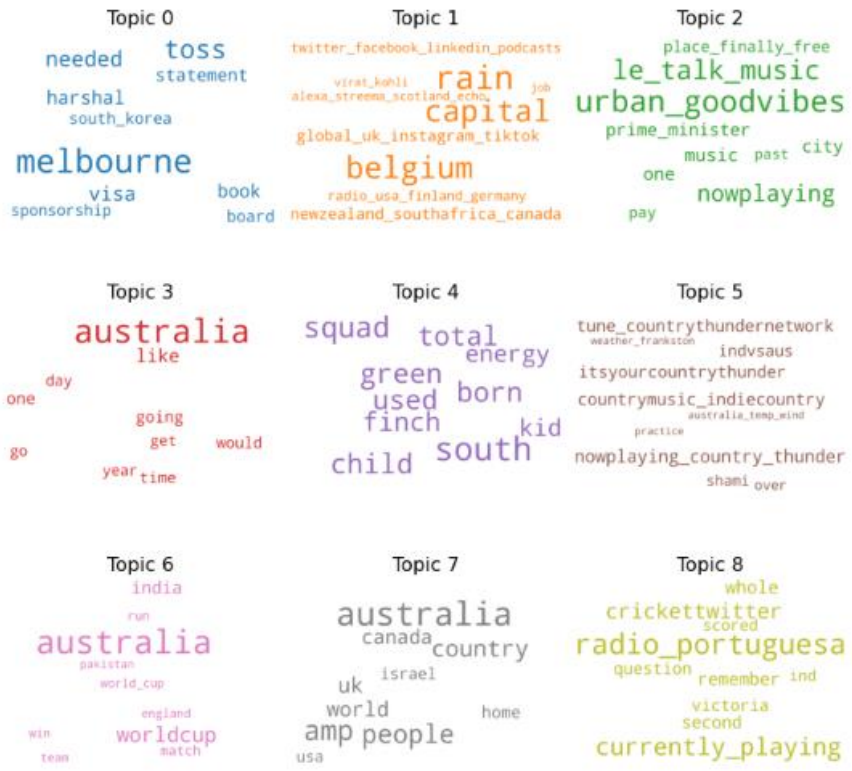

Figure 9: Word cloud for LDA topics

We can see that topics 6 and 8 seems to be mostly related to the recent T20 cricket world cup happening in Australia. Topics 2 and 5 seems to be about a radio station playing music. However, it is still a little difficult to understand what exactly topics 0 and 4 are talking about.

MODEL 2: BERTOPIC

Another model that was implemented was BERTopic model.

An initial clean up of data, such as removing hashtags and urls were done on the data, but any further pre-processing such as removing stopwords, tokenization, lemmatization etc. were not done, since BERTopic uses an embedding approach which depends on maintaining the text's original structure. Word embeddings are a sort of word representation that allows for similar meaning words to have alike representation.

The four main parts of BERTopic are:

- An embedding model: the default embedding model 'all-MiniLM-L6-v2' was used.
- UMAP dimensionality reduction: By default, BERTopic does its dimensionality reduction using UMAP.

- HDBSCAN clustering: After reducing dimensionality, HDBSCAN clustering is used to cluster groups of similar embeddings to extract topics.
- Cluster tagging using c-TF-IDF: Extracting topics for each of the clusters is BERTopic's last phase. BERTopic does this via c-TF-IDF, a modified kind of TF-IDF. c-TF-IDF looks at the most relevant words from each cluster to create topics.

Implementing a basic BERTopic model

At first, a very basic BERTopic model was implemented. For the basic BERTopic model, these steps won't have to be explicitly specified, and its default parameters will be used.

```
model = BERTopic(language="english",calculate_probabilities=True,nr_topics="auto")
```

The number of topics is set to auto.

```
topics,probabilities=model.fit_transform(df1_list)
```

Calling the fit_transform function will take in all the documents and embed it and predict which topic it belongs to.

	Topic	Count	Name
0	-1	17893	-1_and_to_that_are
1	0	1957	0_rain_icc_matches_rainy
2	1	1122	1_indiecountry_countrythundernetwork_countrymusic_itsyourcountrythunder
3	2	967	2_pakistan_pakvsned_pakvszim_zimbabwe
4	3	696	3_kohli_virat_viratkohli_innings
...	...	...	...
545	544	10	544_workers_insecure_apple_striking
546	545	10	545_scovaus_penalty_turnovers_succesful
547	546	10	546_yellow_black_green_uniform
548	547	10	547_nb_mesh_dominik_owc
549	548	10	548_russia_russian_truths_propaganda

550 rows × 3 columns

Figure 10: Topics for BERTopic

Topic -1 here refers to all the outliers without a particular topic and this is not considered. Excluding this we have a total of 549 topics.

Visualizing results

To visualize the topics generated, an intertopic distance map is plotted.

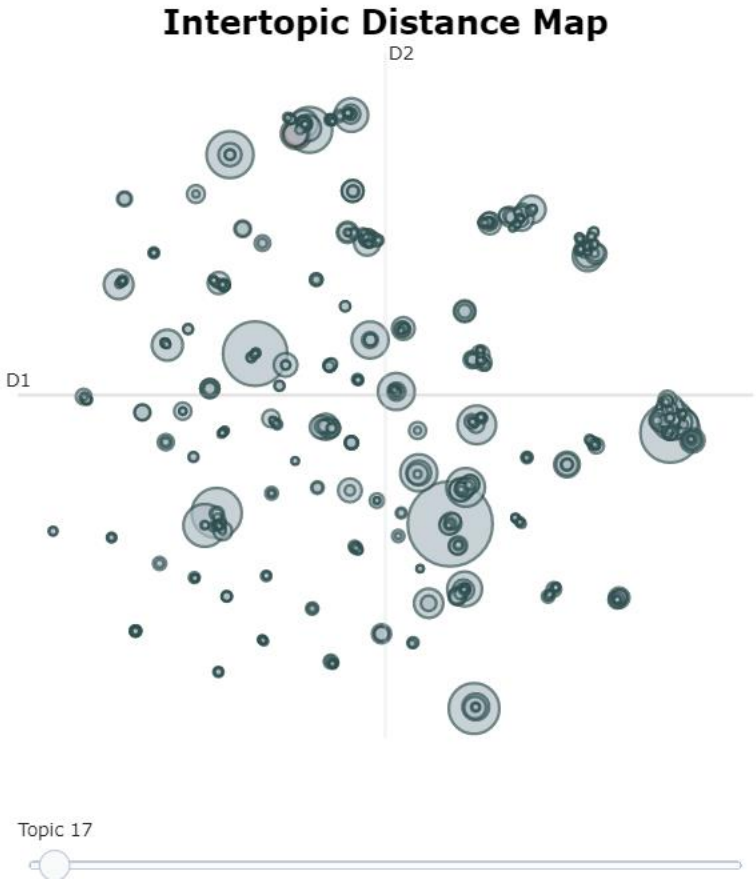

Figure 11: Intertopic distance map

Each circle is a topic, and the size of the circle represents how frequently the topic appears in all texts.

The bar chart uses the c-TF-IDF scores to display the specified terms for a few keywords.

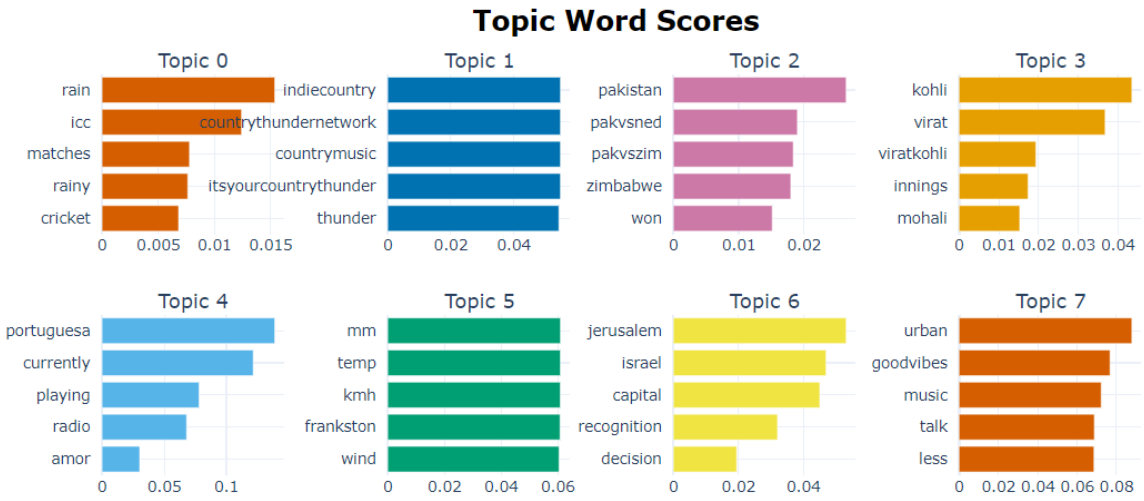

Figure 12: Bar charts for BERTopic topics

The top keywords in topic 0 are ‘rain’, ‘icc’, ‘matches’, ‘rainy’, and ‘cricket’. From this we can deduce that this topic is about a rainy cricket match. Similarly, this can be done with all the other topics generated.

To visualize the similarity between topics, a heatmap is generated.

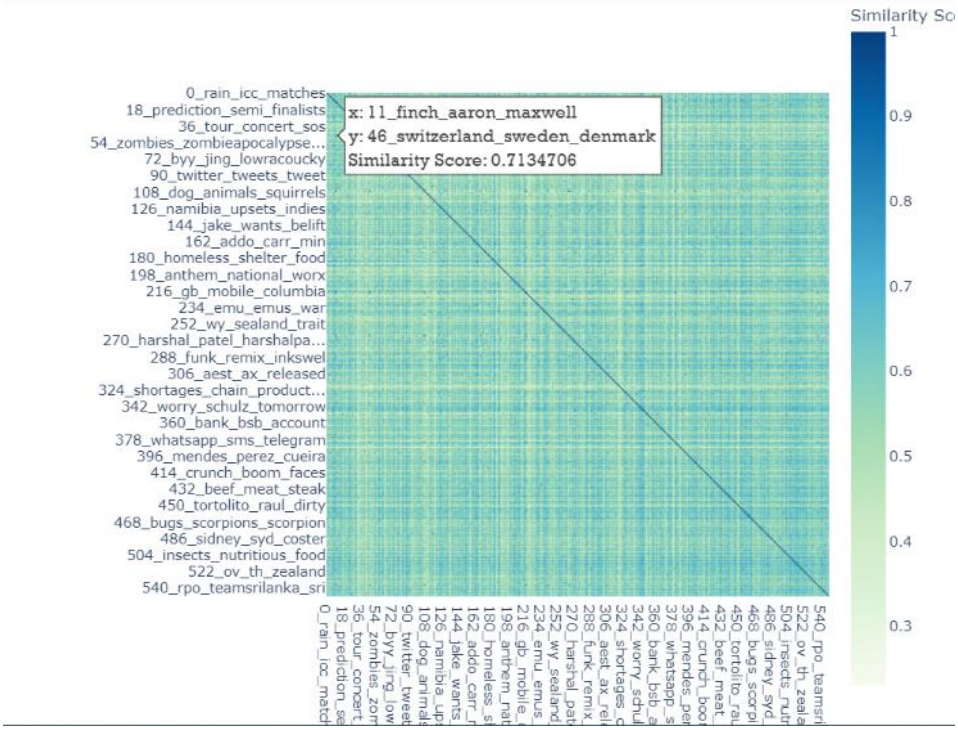

Figure 13: Heatmap for BERTopic

For example, the topics 11 and 46 above have a similarity score of 0.7134706

Reducing topics

Due to the size of the dataset, we can see a lot of topics are generated. We can reduce these topics by setting different parameters, and this will merge similar topics. At the start of this section, we talked about four main steps of BERTopic that uses default settings. Now, we will customize some of these parameters so as to get fewer number of topics.

The hdbscan parameter 'min_cluster_size' is set to a higher value 300 to ensure that the small clusters are not created. This indicated the minimum number of documents to form a cluster.

The umap parameter 'n_neighbors' is also set to a higher value of 200 to avoid formation of those tiny clusters since a cluster needs a lot of nearby documents to form.

```
umap_model = UMAP(n_neighbors=200, n_components=3, min_dist=0.05)
hdbscan_model = HDBSCAN(min_cluster_size=300, min_samples=40,
                        gen_min_span_tree=True,
                        prediction_data=True)
```

Figure 14: Customizing parameters for BERTopic

As part of the c-TF-IDF process, we also initialise a vectorizer model to handle stopword removal.

```
vectorizer_model = CountVectorizer(ngram_range=(1, 2), stop_words=stopwords)
```

After training the model we can see that the number of topics has reduced to 22.

	Topic	Count	Name
0	-1	14429	-1_australia_worldcup_world_india
1	0	17803	0_australia_people_like_us
2	1	1938	1_rain_icc_worldcup_world cup
3	2	1667	2_pakistan_zimbabwe_win_pakvszim
4	3	1557	3_worldcup_australia_england_worldcup australia
5	4	1354	4_zealand_new zealand_new_ausvnz
6	5	1335	5_countrythundernetwork countrymusic_countrythundernetwork_tune countrythundernetwork_indiecountry itsyourcountrythunder
7	6	1113	6_pant_australia_babar_rizwan
8	7	1038	7_urban_music_less talk_talk music
9	8	956	8_finch_maxwell_worldcup_australia
10	9	900	9_india_england_south africa_africa
11	10	709	10_portuguesa_portuguesa australia_radio portuguesa_currently playing
12	11	593	11_kohli_virat_virat kohli_innings
13	12	591	12_jerusalem_israel_capital_recognition
14	13	585	13_floods_australia_weather_rain
15	14	512	14_radio_mintwave_mintwave radio_southafrica canada
16	15	501	15_india_shami_warm_warm match
17	16	480	16_group_ireland_canada_nigeria
18	17	454	17_sri_lanka_sri lanka_srilanka
19	18	423	18_scotland_rugby_rlwc_scotland australia
20	19	408	19_netball_netball australia_gina_sponsorship
21	20	328	20_mm conditions_australia temp_weather frankston_wind kmh
22	21	326	21_stoinis_marcus_marcus stoinis_fastest

Figure 15: New topics generated after reduction

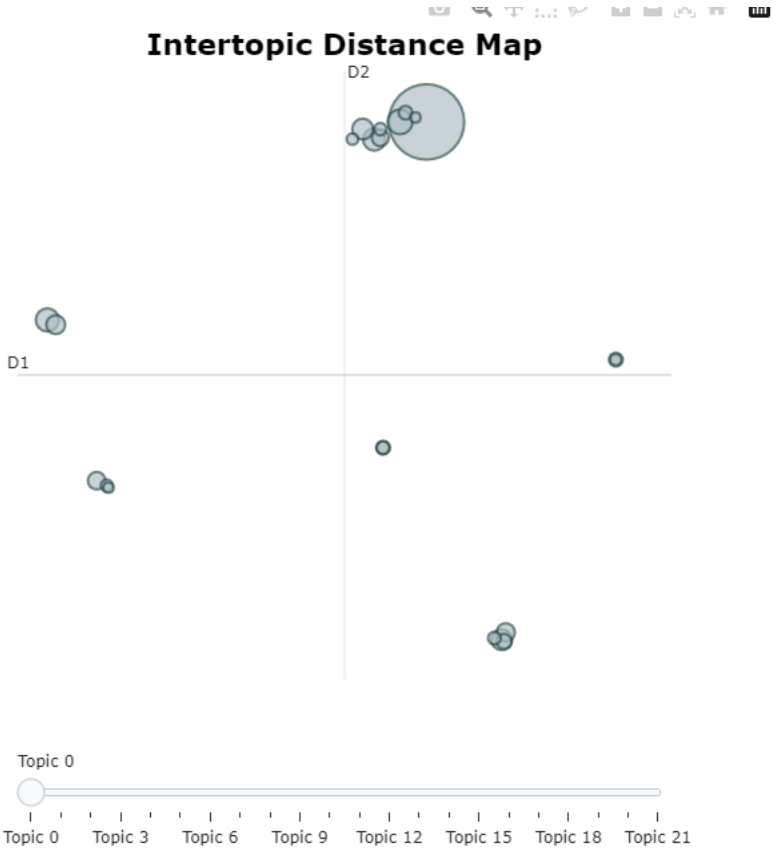

Figure 16: Intertopic distance map for reduced topics

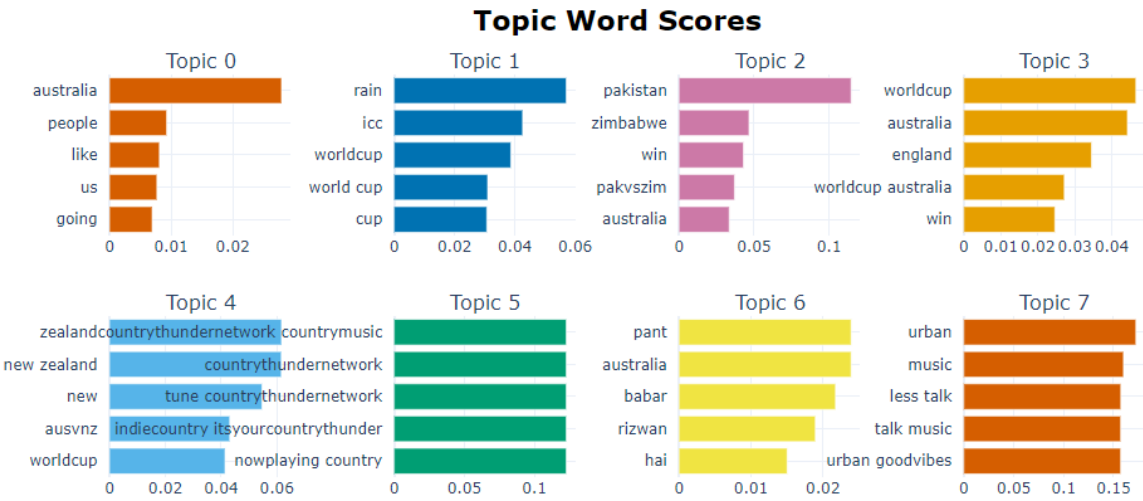

Figure 17: Bar charts for reduced topics

Here, the bar graphs of the first 8 topics are displayed. Since the T20 world cup has been taking place in Australia, and the latest 50,000 tweets based on Australia were extracted, we can see

that majority of the topics are related to this, while some other topics are related to a radio station playing songs.

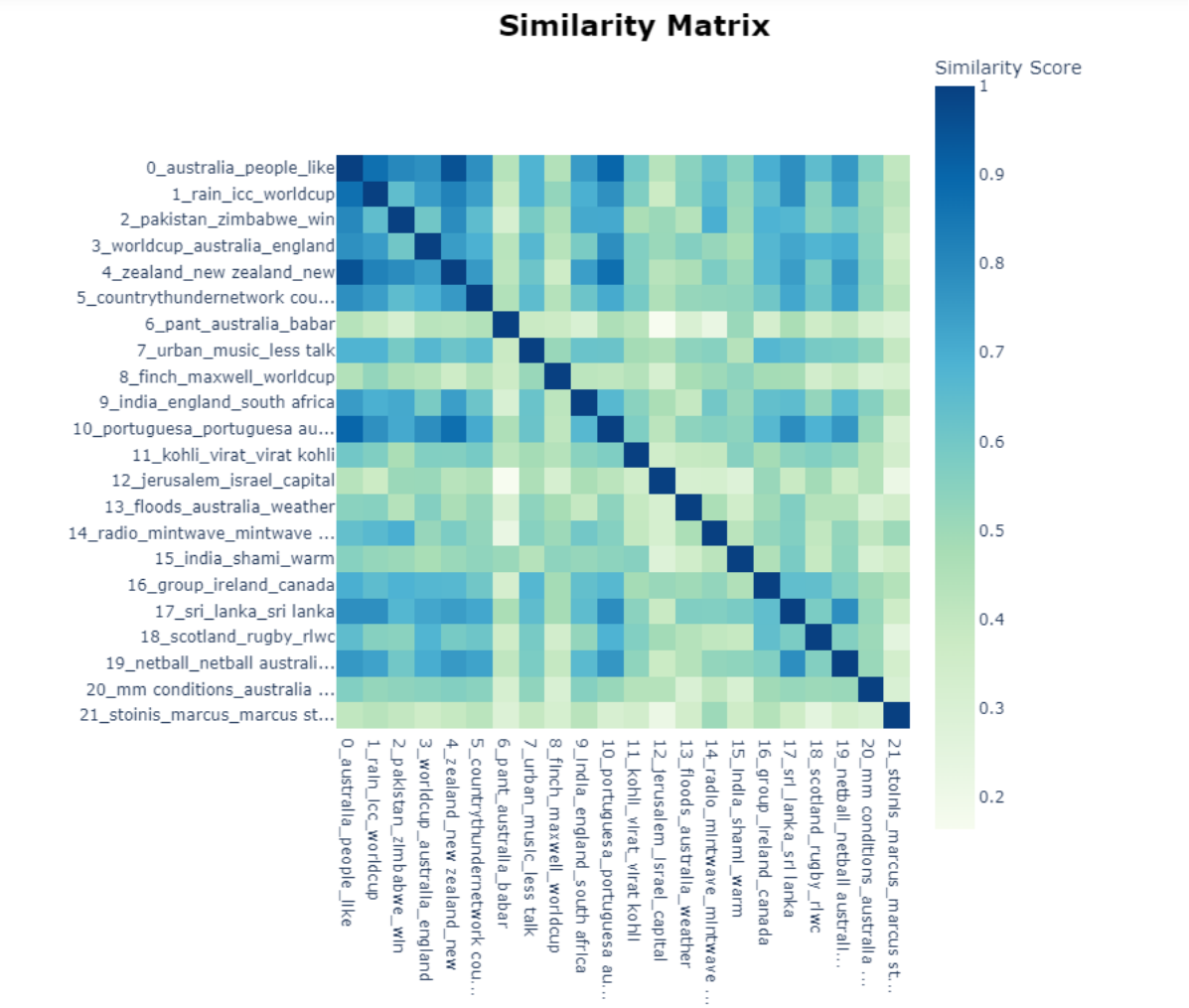

Figure 18: Heatmap for reduced topics

COMPARISON

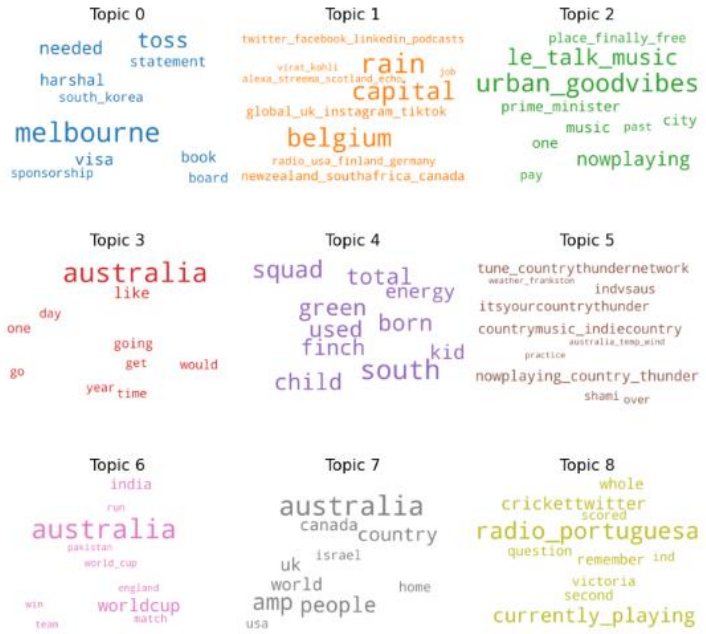

Figure 19: Word cloud for LDA topics

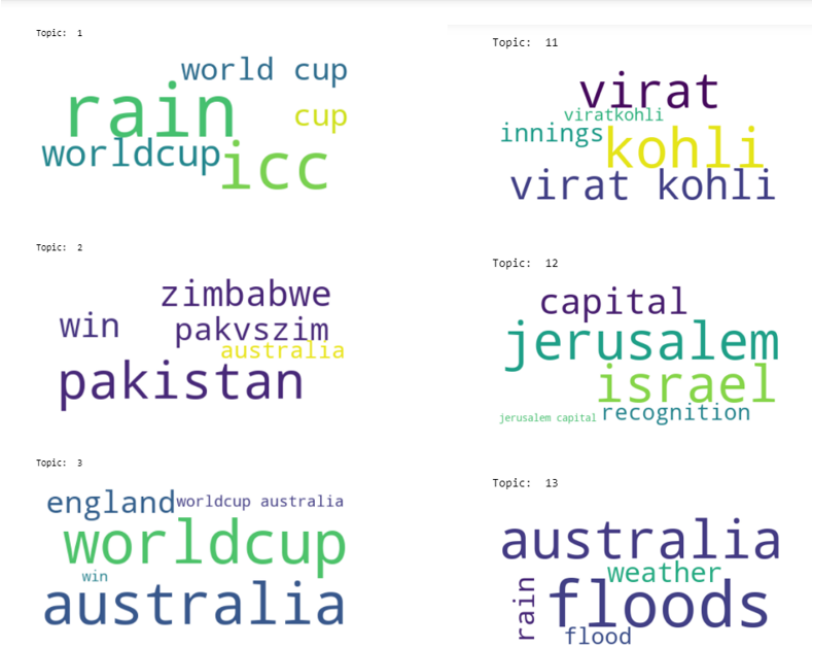

Figure 20: Word cloud for BERTopic topics

When looking at the topics generated by both LDA model and BERTopic, we can see that BERTopic model gives more easily understandable and distinct topics as compared to LDA.

We can see in topic 8 for LDA (Figure 19), there are keywords relating to cricket(crickettwitter, score) and the radio station(currently_playing), whereas in BERTopic, these are considered different topics. This could be because unlike the bag-of-words method, high-quality embeddings take into consideration the semantic link between words in a corpus. This results in more interesting and worthwhile topics, and due to the semantic nature of embeddings, pre-processing is not required in most cases. However, BERTopic did take a lot longer to train in comparison to LDA.

One major difference in LDA is that we must specify the number of topics beforehand, but this is not required in BERTopic.

CONCLUSION

This project focused on performing topic modeling on 50,000 tweets based in Australia, and comparing the topics generated by two models. While the first model LDA is the standard model used in topic modeling, the second model BERTopic is a newer, less explored model. On comparing the topics generated by both models, although LDA did produce decent topics, BERTopic produced more easily interpretable topics which helps the user easily understand what was being discussed in the tweets.

The code for the project can be accessed through the following link: Click here