Natural Language Processing

TOPIC MODILING

Team ID: **T037**

INTODUCTION

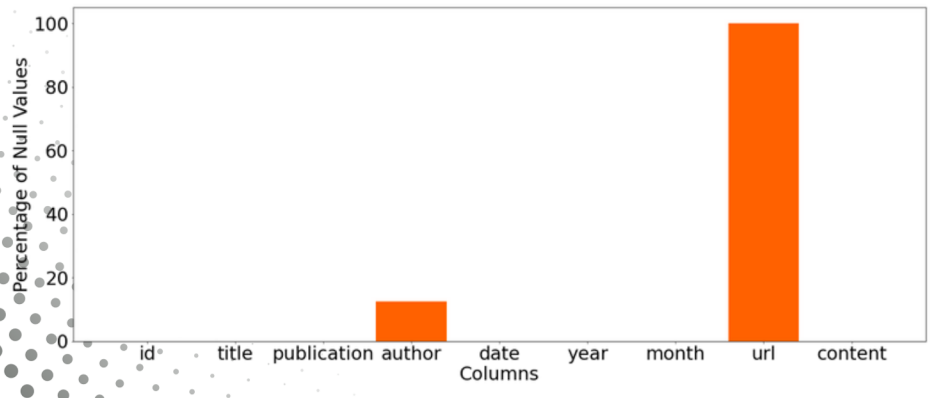
First, we need to find a **group of words** that **identify the topic** of the document, to do this task we need to apply preprocessing to improve the **quality**, **consistency**, and **efficiency** of the model. By applying preprocessing techniques to text data, we can reduce noise, normalize the data, improve consistency, and make it easier for the topic modeling algorithm to identify meaningful topics. for this, we **handle the missing values** in columns and rows, **handle columns data types**, deal with **unnecessary columns**, remove **HTML tags**, **URLs**, **punctuations**, **special characters**, **unnecessary words**, **stopwords**, **frequent words**, and **rare words** then we apply **stemming** and **lemmatization** and apply **POS tagging** in **tokenized words** then we deal with **categorical data (Nominal)** and finally Extract feature using **TF-IDF** and **K-Means** to get a group of words that identify the topic.

DATA CLEANING

1- Handel Missing values.

- Handel columns with missing values :
  - By Getting the total number of nulls in each column we found that we should **drop URL column** because it's empty(100% null values) as showing here

id	0.000
title	0.000
publication	0.000
author	12.612
date	0.000
year	0.000
month	0.000
url	100.000
content	0.000



DATA CLEANING

1- Handel Missing values.

- Handel rows with missing values :
  - By getting the total number of rows containing missing values we should **remove all rows** containing null values because It represents a small portion of the data, about **12.6%** of the data

```
id             0.0
title          0.0
publication    0.0
author         0.0
date           0.0
year           0.0
month          0.0
content        0.0
```

2- Handel columns data type.

- By getting the data type of each column we found some columns have data type **object** as showing

```
id              int64
title          object
publication    object
author         object
date           object
year          float64
month         float64
content        object
```

- So that we **change** their data type to the right one as showing

```
id                        int64
title                    string
publication              string
author                   string
date           datetime64[ns]
year                      int64
month                     int64
content                  string
```

3- Dealing with unnecessary columns.

- By Examiner the data we found:
  - the **id column** is unique and should be **dropped** because it can be safely removed without affecting the quality or accuracy of the analysis or modeling task.

```
title                    string
publication              string
author                   string
date           datetime64[ns]
year                      int64
month                     int64
content                  string
dtype: object
```

DATA CLEANING

3- Dealing with unnecessary columns.

- By Examiner the data we found:
  - the **date column** makes data redundancy **extracting day** from date column and **renaming the column from date to day**

date	year	month
2016-12-31	2016	12
2017-06-19	2017	6
2017-01-06	2017	1

day	year	month
31	2016	12
19	2017	06
06	2017	01

DATA PREPROCESSING

1- Convert to lowercase.

- To apply preprocessing in text data we should to convert all **string** columns [*title*, *author*, *content*] to lowercase.

2- Remove HTML Tags.

- we should remove HTML tags from columns [*title*, *author*, *content*] by **replacing** the text that matches the regular expression (**<.*?>**) with an **empty string**.

3- Remove URLs.

- we should remove URLs from columns [*title*, *author*, *content*] by **replacing** the text that matches the regular expression (**https?://\S+www\.\S+**) with an **empty string**.

DATA PREPROCESSING

4- Remove Punctuation

- we should remove punctuations from columns [*title*, *content*] by using **string.translate()** and it's a method that can be used to replace punctuations in a string with an **empty string**.

5- Remove Special Characters

- we should remove special characters from columns [*title, content*] by **replacing** the text that matches the regular expression (**[^a-zA-Z0-9]**) with a **single space**.

6- Remove *unnecessary spaces* and words

- after removing the preceding characters we found that strings contain extra spaces so we apply the regular expression (**\s\s+**) to **replace** them with a **single space**
- after examine the data we found that **title column** contains **publication** at the end of text and it is considered redundancy data so we removed it by making string matches a regular expression (**.**publication**$**) and replacing string by an **empty string**.

DATA PREPROCESSING

7- Apply Tokenization.

- Apply sentence tokenization:
  - the first step in tokenization is to tokenize string on [*title*, *content*] columns by sentences using **sent_tokenize** in **NLTK**
- Apply word tokenization:
  - the second step in tokenization is to tokenize sentences on [*title*, *content*] columns by words using **word_tokenize** in **NLTK** .

8- Remove Stop words.

- after tokenization, we remove stop words like "the", "a", "an", and "and". from columns [*title*, *content*] to Reduce noise and improve accuracy because it considered to be low-information words that do not convey much meaning.

9- Remove Frequent Words.

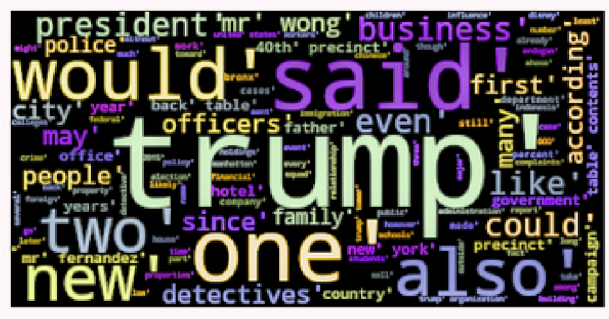
- we remove **The most frequently 10 words** in columns [*title*, *content*] to reduce noise and improve efficiency because it is considered noise if they are not relevant to the specific NLP task.

DATA PREPROCESSING

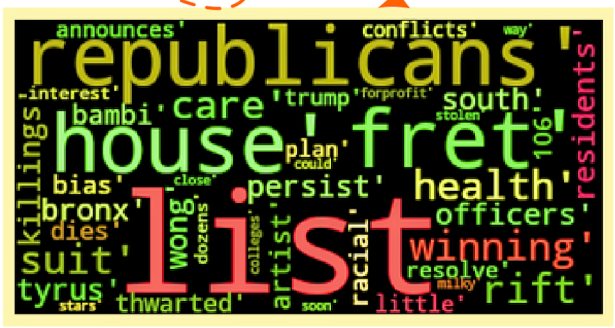
10- Remove Rare Words

- we remove **The most 10 rare words** in columns [*title*, *content*] to reduce noise and improve efficiency because it is considered to be noise if they occur very infrequently and do not convey much meaning or context.

Word Cloud



content column



title column

11- Apply Stemming

- we apply stemming using the **Porter stemming** algorithm in columns [*title*, *content*] to reduce words to their base form so that different forms of the same word are treated as the same

12- Apply POS Tagging and Lemmatization

- we apply lemmatization using the **wordnet lemmatizer** and apply POS Tagging in columns [*title*, *content*] by mapping (NOUN, VERB, ADJECTIVE, ADVERD) to reduce words to their base form so that different forms of the same word are treated as the same and solve disambiguation by analyzing the context of a word and its part of speech.

DATA PREPROCESSING

13- Dealing with categorical data (Nominal Data).

- Preprocess [publication]:
  - we get the number of unique values in publication column and found it = 5 so the best way to deal with it that applying **one-hot-encoding** to convert *text data* to *structured data*.

```
title                              object
author                             object
day                                string
year                               string
month                              string
content                            object
publication_atlantic                uint8
publication_breitbart               uint8
publication_business insider        uint8
publication_cnn                     uint8
publication_new york times          uint8
dtype: object
```

- Preprocess [author]:
  - we get the number of unique values in author column and found it have 2351 unique values so the best way to deal with it that **mapping** authors' names to convert *text data* to *structured data*.
  - to get number of unique values we cleaned data by:
    - remove special characters and unnecessary words
  - separated authors in cell
  - calculate frequency for each author
  - reduced authors to only one author
  - mapping column by values from 0 to 2351

After Listing

	author
0	[carl hulse]
1	[benjamin mueller, al baker]
2	[margalit fox]
3	[william mcdonald]
4	[choe sang-hun]
5	[sewell chan]
6	[javier hernández]
7	[gina kolata]
8	[katherine rosman]
9	[newman]
10	[justin gillis]

After Mapping

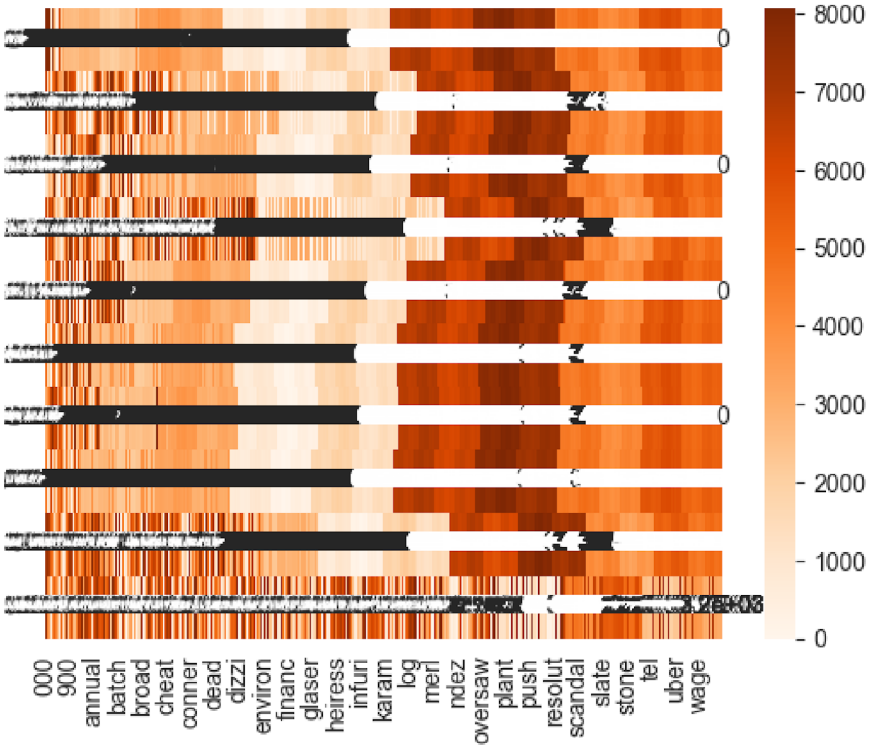
	author
0	0.0
1	1.0
2	3.0
3	4.0
4	5.0
5	6.0
6	7.0
7	8.0
8	9.0
9	10.0
10	11.0

FEATURE EXTRACTION

- **Apply TF-IDF.**
  - we apply TF-IDF to transform raw input data into a format that is suitable for use in a machine-learning (**K-Means**) model.

MODEL TRAINING AND TESTING

- **Apply K-Means.**
  - we apply K-Means to group similar items together based on their features.
  - K-means clustering is used to identify the most important topics in a text corpus. By clustering together documents that contain similar words or phrases.



The heatmap for 100 record