ASSIGNMENT 3

Understanding the Dataset –

The dataset used in the report is a 3 Million troll tweets sent from Twitter handles connected to the Internet Research Agency, a Russian Troll factory. The tweets are collected during the timeline February 2012 and May 2018 to create this dataset. Out of 3 Million tweets, 2.11 Million tweets were sent out in English and other 900K tweets in different languages like Russian, Hindi, Arabic, Spanish, etc. These 3 Million tweets are divided into 13 .csv files. Each file contains 20 columns containing the information like author handle, tweet content, language of the tweet, region from which tweet is posted, time at which tweet is posted, tweet ID, retweet binary indicator, account category, etc.

Overview of the Report –

This report is designed on the foundation of text data modelling concepts. We are using word embeddings techniques like word2Vec model of genism module in python to convert the corpus into a set of vectors and identify the similarities and biasness of the data.

To apply all the text modelling concepts we'll first need to pre-process the data to remove any scalability and quality issues while doing the operations on it. Pre-processing of this raw data is very important because working on large dataset without any pre-defined set of models poses scalability and data quality problems which further reflects in the analysis.

After the pre-processing, we'll try to identify the biasness in the language used in the tweet data. Using word embeddings, we'll compute the Euclidean similarity in the tweets and identify the biased opinions of the authors of the tweets.

After analysing the biasness in the data, we'll try to apply some relational and temporal modelling concepts to identify the number of tweets posted for each account category and the number of tweets posted over the timeline of around 6 years of the dataset.

Lastly, we'll apply clustering operations and try to identify the accuracy of content similarity alignment done by the twitter APIs with the corresponding account category labels provided such as 'RightTroll', 'LeftTroll', 'Hashtagger', 'Commercial', etc, as seen in the relational and temporal modelling section.

We are using Python and Tableau as the major tools for the report. Modules and packages of python used in this report are Pandas, RE, NLTK, ITERTOOLS, CONTRACTIONS, GENSIM, NUMPY, SKLEARN, MATPLOTLIB.

Data Pre-processing -

Data pre-processing is a pivotal step to enhance the quality of the raw dataset in order to extract useful insights from it without encountering any quality issues. It's a technique of sanitizing the raw data to make it suitable for text data models. We use the following methods to sanitize our data.

- *Removal of all Non-English tweets –*

  We removed all the tweets which are not made in English language because there are 2,116,867 English tweets in our dataset as opposed to few thousand tweets of all the other languages which is insignificant for our analysis. Hence, we removed all the other language tweets from our dataset for the data analysis.

- *Removed URL Links –*

  As mentioned in the README file of the dataset that URLs in the contents might be active links and may lead to some undesirable content. So, we have removed the URLs from all the tweets. Removing anything starting from 'http\S+' till a space in the text is encountered.

- *Lowercase the text –*

  Converting all the uppercase letters in tweets to the lower-case format which will help us scale the data to identify biasness. For example, some of the tweets in the data has the word 'PresiDenT' or 'TrumP' in it which hampers our calculations for biasness. Hence, we reduced all the words to lowercase for better accuracy.

- *Contractions removal –*

  Removing all the contractions from the data and expanding it into the original form. For example, writing I'd as I would, I'll as I will and so on. This will help in text standardization and better analysis.

- *Standardizing words –*

  It is a process of transforming a text into normal standard form. For example, 'gooood' or 'gud' transformed into 'good' which is the correct word for both. This helps in removal of misspellings and out of vocabulary words.

- *Removing Punctuations –*

  Removing of unnecessary integers, float values, and special symbols from the text data helped us in standardizing the tweets contents. This was done because the file number 6 was unable to go through the text pre-processing as it had unnecessary float values in the tweets' contents. Hence, we removed it from the file. In figure 1 below we can see the error encountered without removing punctuations.

```
  File "C:\Users\risha\anaconda3\lib\site-packages\pandas\core\series.py", line 3848, in apply
    mapped = lib.map_infer(values, f, convert=convert_dtype)

  File "pandas\_libs\lib.pyx", line 2329, in pandas._libs.lib.map_infer

  File "<ipython-input-315-1b55f564a932>", line 2, in <lambda>
    lambda x: x.lower())

AttributeError: 'float' object has no attribute 'lower'
```

Fig. 1

- *Remove Stop-words –*
  A set of commonly used words in a language are known as stop-words. Example of English stop-words are ‘a’, ‘the’, ‘this’, ’is’, ’are’, etc. These are low information words and removal of these words divert the focus of analysis towards high frequency words. Also, it helps in controlling the size of model.

- *Lemmatizing the words –*
  The removal of inflections and mapping the word to its root form is known as lemmatization. It is very similar to stemming but it actually transforms the word into the real root form which is present in the dictionary as opposed to chopping of words in the stemming. Hence, we have used lemmatization in place of stemming.

In figure 2, is the code used for all the above pre-processing tasks. All the lambda functions are run on all 13 files to get ourselves a data worthy of analysis.

```
                        # Remove punctuations
punctuations_regex="!#%&\'()\*+,-/:;<=>?@\[\]^_`{|}~.1234567890"
df6[column] = df6[column].apply(
             lambda x: re.sub('[%s]'%re.escape(punctuations_regex), ' ', x))

                          # Lowercase
df6[column] = df6[column].apply(lambda x: x.lower())

                          # Remove links
df6[column] = df6[column].apply(lambda x: re.sub(r'http\S+', '', x))

                        # Removal of Contraction in the words
df6[column] = df6[column].apply(lambda x: contractions.fix(str(x)))

                        # Standardizing the words
df6[column] = df6[column].apply(lambda x: ''.join(''.join(c)[:2]
                                                  for _, c in itertools.groupby(x)))

                        # Removal of stop-words
df6[column] = df6[column].apply(lambda x: remove_stopwords(x))

                        # lemmatize_word(text)
df6[column] = df6[column].apply(lambda x: lemmatize_word(x))

            #Reframe as Strings to create Word2Vec model
df6[column] = df6[column].apply(lambda x: ' '.join(map(str, x)))
```

Fig. 2

Remove_stopwords() and lemmatize_word() are the function calls made to perform operations separately.

BIAS Identification –

Data bias is a concept in which some elements of the dataset are heavily represented and outshining other elements. There are many ways a bias can occur in a text database, like Gender, Stereotypical,

Racial, Religion bias, etc. In our dataset we used word2Vec models to find out some of the biases listed below with the similarity as calculated from the trained model.

- *Gender Political bias –*

  From our dataset we can identify that there was gender biasness in the people of the world related to the political presidential elections of the United States. We applied word embeddings and realized that more people used the term 'president' in context with 'mr' as compared to 'madam'. Since we know, in the presidential race of 2017-2021 term, Donald Trump was running against Hillary Clinton and Barack Obama was leaving the office after his term completion. In our model, we computed similarity of the words' 'trump', 'obama', and 'hillary' in the context of 'president' to see the Euclidean similarity of them. In figure 3 below we can see the values of Euclidean similarity that are computed.

```
In [273]: model3.wv.similarity('mr','president')
Out[273]: 0.41249487

In [274]: model3.wv.similarity('madam','president')
Out[274]: 0.27270746

In [275]: model3.wv.similarity('trump','president')
Out[275]: 0.6157454

In [276]: model3.wv.similarity('obama','president')
Out[276]: 0.5069431

In [277]: model3.wv.similarity('hillary','president')
Out[277]: 0.42390624

In [278]: model3.wv.similarity('donaldtrump','president')
Out[278]: 0.49699944

In [279]: model3.wv.similarity('hillaryclinton','president')
Out[279]: 0.37551153
```

Fig. 3

As we can clearly see from figure 3 that the people who tweeted about the elections or related to the elections clearly mentioned the word president frequently with the male candidate (**0.412, 0.615, 0.506**) as compared to the female candidate (**0.272, 0.423, 0.375**). Even the retiring male candidate of the office is more famous on twitter and have more mentions with the word president as compared to the female candidate. This is a gender bias on the grounds of political front seen in the dataset. The popularity for trump seen in these tweets might be one of the reasons of his successful campaign.

- *Stereotypical Racial Bias –*

  From our dataset we can identify a stereotypical racial bias in the tweets posted from all over the world. We used our model to find similarity between the words 'policebrutality', 'white',

'black', 'whiteamerica', and 'blackamerica' in order to seen whether the notion floating around the world related to the people of colour is true or not.

From our computations we can see that the association of 'policebrutality' with the words like 'black' **(0.306)** and 'africanamerican' **(0.401)** are more as compared to the words like 'white' **(0.195)** and 'whiteamerica' **(0.191).** This works on the grounds of stereotypes all over the internet that the black community of America is targeted and racial profiled by the police.

Also, from computations in figure 4, we can see the top most associated word with 'policebrutality'. The words like 'btp', 'blackmatters', 'black-skin-is-not-a-crime', 'baltimore-vs-racism' can be categorized as the racial biasing of the tweets posted all around the world.

```
In [319]: model2.wv.most_similar('policebrutality')
Out[319]:
[('acab', 0.8539137840270996),
 ('btp', 0.8493797183036804),
 ('dontcall', 0.7935729026794434),
 ('blackmatters', 0.7698806524276733),
 ('ilghhes', 0.7671732902526855),
 ('antipolicebrutalityday', 0.766614556312561),
 ('blackskinisnotacrime', 0.7581493854522705),
 ('heroesinblue', 0.7554571628570557),
 ('policeviolence', 0.7544670104980469),
 ('baltimorevsracism', 0.7483412027359009)]

In [320]: model2.wv.similarity('policebrutality','white')
Out[320]: 0.19514924

In [321]: model2.wv.similarity('policebrutality','black')
Out[321]: 0.30164286

In [322]: model2.wv.similarity('policebrutality','whiteamerica')
Out[322]: 0.19122587

In [323]: model2.wv.similarity('policebrutality','africanamerican')
Out[323]: 0.40168858
```

Fig. 4

- *Religion based Bias –*

From our dataset we can identify any religion biased reflecting on the tweets posted all around the world. We used our models to find the most associated words with certain religions like 'Islam', 'Christianity', 'Judaism', 'Buddhism' and 'Hinduism' in the hope of seeing some specific stereotypical words that are affiliated with the people from a certain type of religion.

When we compute the top 10 associated word with 'islam', we can clearly see that the media fears and unwanted reflections of the religion has resulted in a stereotypical religion- based bias in the world, and same reflects from our dataset.

The words like ‘jihad’ and ‘barbaric’ are two of the top words with ‘islam’ which tends to tell an unpleasant story about the religion that has been created in the world due to some stereotypical news broadcasts from the main stream media.

Also, we can see from below figure that the word ‘islam’ and ‘christianity’ are the top words in each other lists but if we compare the top 10 words of both religions, we will see big difference. We can see that the words like ‘morality’ and ‘tolerance’ are in the top 10 words list of ‘christianity’.

We can also see the word ‘barbaric’ is associated with ‘christianity’ as well but it can be argued about that it is due to the fact that ‘christianity’ and ‘islam’ have occurred together in almost 80% of the similar tweets. Also, there are no positive words associated with ‘islam’ which is not the case with ‘christianity’ which grounds for a religion bias in our tweets’ dataset.

```
In [264]: model3.wv.most_similar('islam')
Out[264]:
[('christianity', 0.808750331401825),
 ('religion', 0.7662930488586426),
 ('muslims', 0.747567892074585),
 ('ideology', 0.7351172566413879),
 ('islamism', 0.7278927564620972),
 ('radical', 0.7180903553962708),
 ('extremism', 0.7167695760726929),
 ('jihad', 0.7155132293701172),
 ('barbaric', 0.7123334407806396),
 ('islamist', 0.7089287042617798)]

In [265]: model3.wv.most_similar('christianity')
Out[265]:
[('islam', 0.8087501525878906),
 ('ideology', 0.7404614686965942),
 ('islamists', 0.7342816591262817),
 ('barbaric', 0.7334513664245605),
 ('islamism', 0.7323970794677734),
 ('morality', 0.7069594264030457),
 ('jews', 0.7061941623687744),
 ('judaism', 0.7016005516052246),
 ('christians', 0.7006170153617859),
 ('tolerance', 0.6985112428665161)]
```

Fig. 5

Moreover, if we compute the top 10 words related to religions like ‘judaism’, ‘buddhism’ and ‘hinduism’ we’ll see a religion-based biasness with the help of our model.

The top words related to ‘judaism’ are ‘incompatible’, ‘totalitarian’, ‘masquerade’ which kind of paints a negative picture about the people of the religion. But also, the word ‘religion-of-peace’ is associated with this religion which portrays positive aspect as well.

```
In [266]: model3.wv.most_similar('judaism')
Out[266]:
[('incompatible', 0.7654170393943787),
 ('abhor', 0.7591867446899414),
 ('abrahamic', 0.7483289837837219),
 ('legitimize', 0.7414585947990417),
 ('antichrist', 0.7403252124786377),
 ('totalitarian', 0.736809492111206),
 ('ctbbyg', 0.7329310178756714),
 ('masquerade', 0.7319992780685425),
 ('ziucdkjbg', 0.7261830568313599),
 ('religionofpeace', 0.7214301824569702)]

In [267]: model3.wv.most_similar('buddhism')
Out[267]:
[('slds', 0.7833845615386963),
 ('tweetjukebox', 0.7768365144729614),
 ('kyzbmabsk', 0.764119029045105),
 ('wordengineer', 0.7625745534896851),
 ('mcmillenhiggins', 0.7540634274482727),
 ('taoism', 0.753142237663269),
 ('quotessphere', 0.7530364990234375),
 ('quoteologist', 0.7369089126586914),
 ('magruderr', 0.7357711791992188),
 ('btwashington', 0.7353904247283936)]
```

Fig. 6

The words like ‘taoism’ and ‘quoteologist’ tends to indicate a bias based on religion with ‘buddhism’. These words portray the peacefulness and devotion in the ‘buddhism’ religion which creates the grounds of biasness.

The words like ‘indoctrinate’, ‘apologist’ and ‘incompatible’ associated with the religion ‘hinduism’ illustrates a biased image about a religion.

```
In [268]: model3.wv.most_similar('hinduism')
Out[268]:
[('ziucdkjbg', 0.7349545955657959),
 ('indoctrinate', 0.7304336428642273),
 ('religionofpeace', 0.7199417352676392),
 ('judaism', 0.7117290496826172),
 ('oinc', 0.7084929943084717),
 ('khdrx', 0.699295163154602),
 ('xpht', 0.6959681510925293),
 ('apologist', 0.6945617198944092),
 ('incompatible', 0.6919213533401489),
 ('oldpicsarchive', 0.6900175213813782)]
```

Fig. 7

Word Embedding Visualization using T-SNE –

T-SNE is the tool used to map high dimensional text data into a 2D plane which indicates the distance between the closest words from our used word.

We have used this visualization technique to see the mapping of closest words from ‘trump’. In figure 8 below we can see indication of the top 10 most similar words in the dataset with the word ‘trump’.

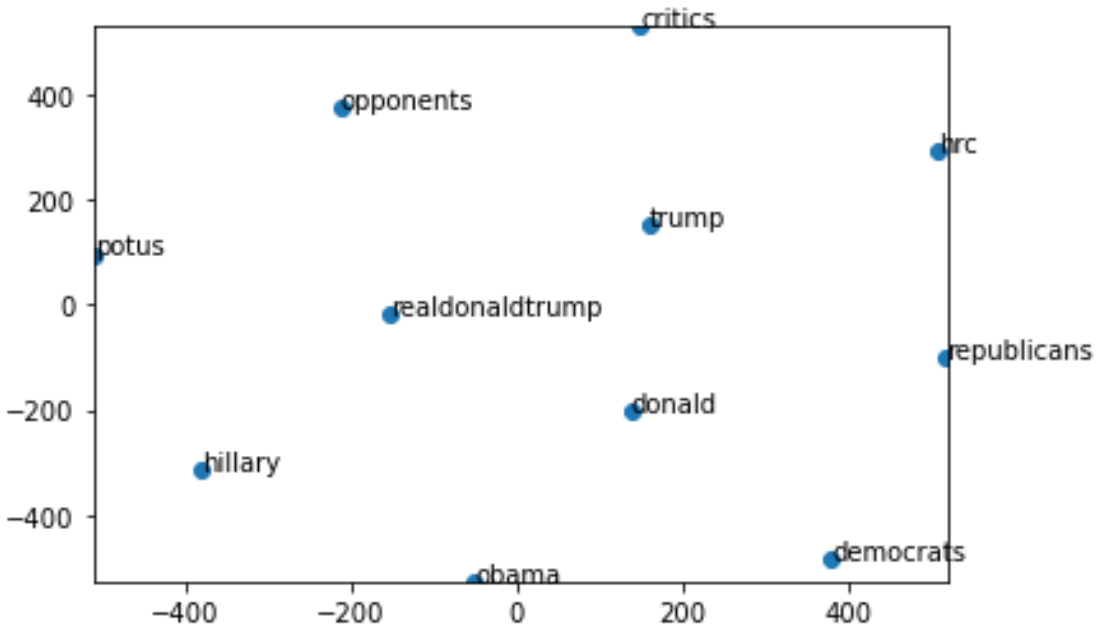

Fig. 8

Figure 9 below is the mapped visualization of the top 10 similar words related to the word ‘policebrutality’ in our dataset. As we can see that the closest words to it are the same words seen above in the racial bias section which proves the racial biasness in our dataset.

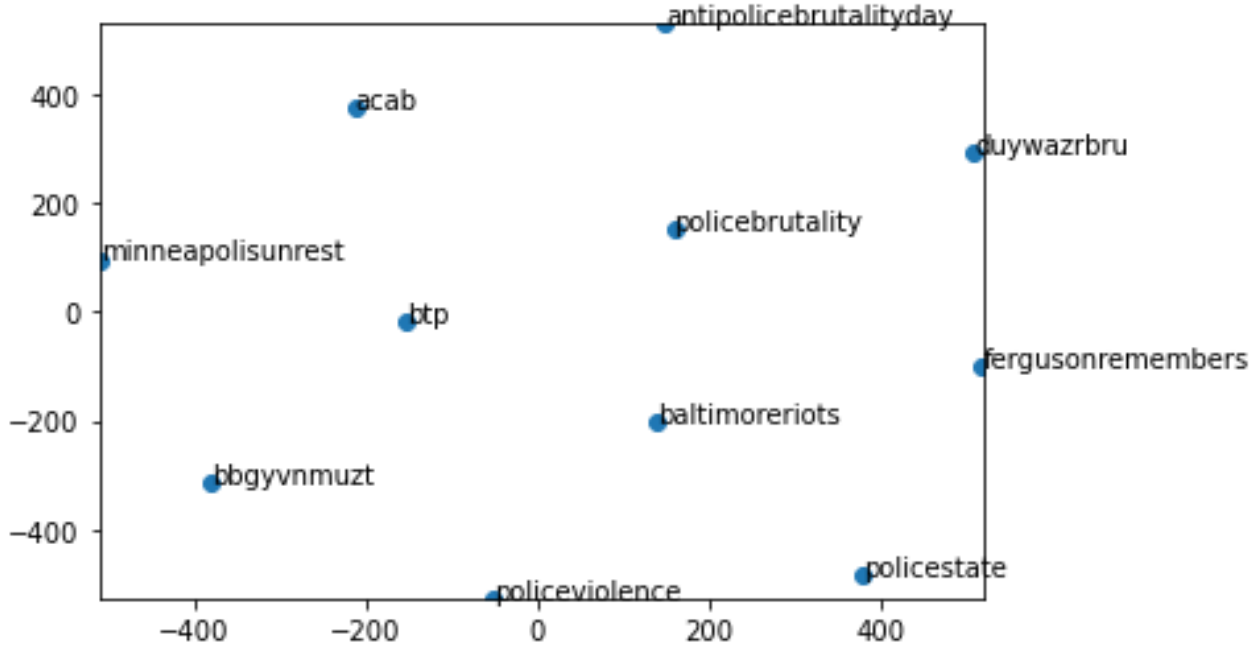

Fig. 9

Tweets per Account Category –

In the dataset of 2.11 Million English tweets, twitter has categorized the accounts in categories with respect to the tweets that those accounts have posted.

Twitter uses their text data models and devise a model to compute similarity between the posted tweet with respect to the default tweet they have assigned for each category, and if the more than average number of a particular categorical tweets are posted from an account then that account is labelled in one of the account categories.

In figure 10 below, we can see the pictorial bar graph representation of number tweets categorized as the standard categories by the twitter.

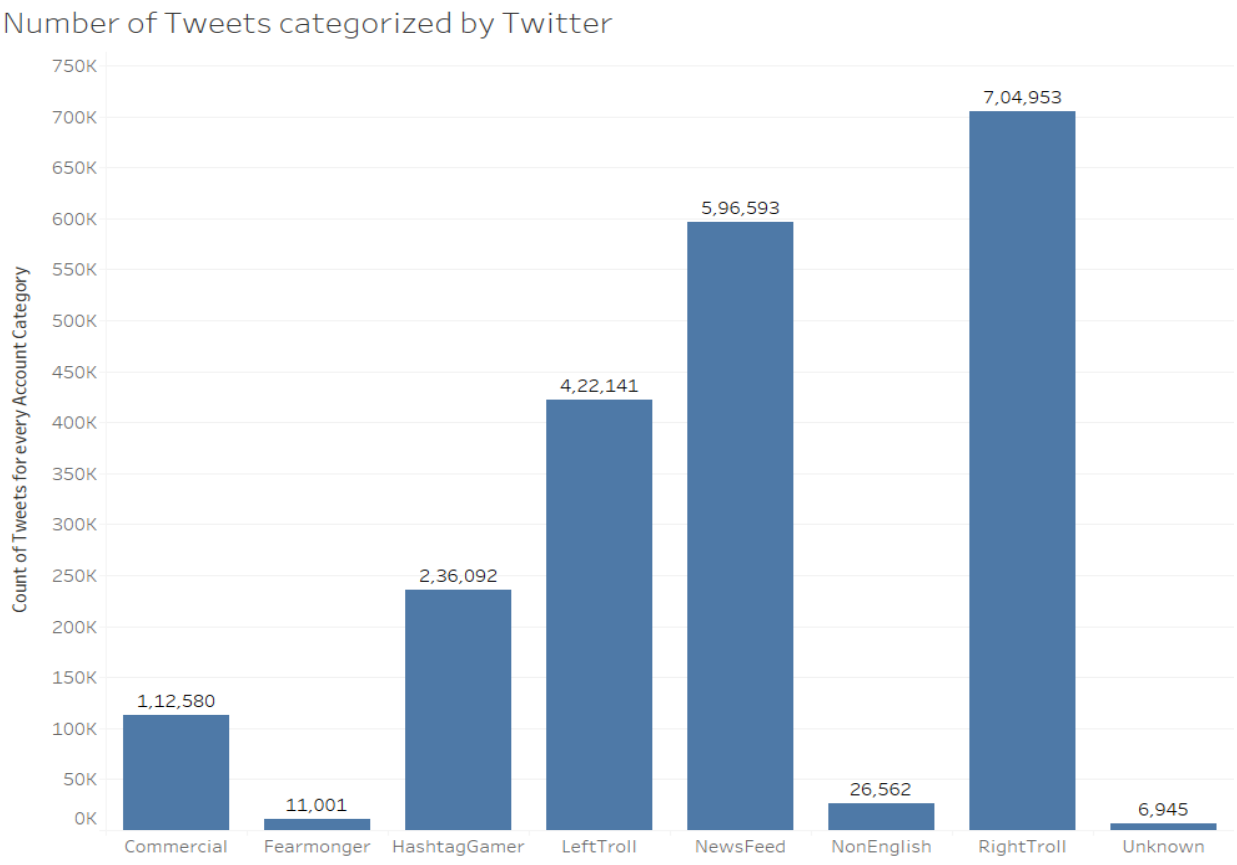

Fig. 10

We used relational modelling concepts to group all the tweets as per account category and then calculate the number of tweets categorized by twitter in each category.

Then, we applied temporal modelling concept to identify the timeline during which these tweets were posted.

We concatenated all the 13 .csv files in one file, changed the published date into a readable published year and then we mapped the count of tweets made every year in all the categories we see in above figure using Tableau.

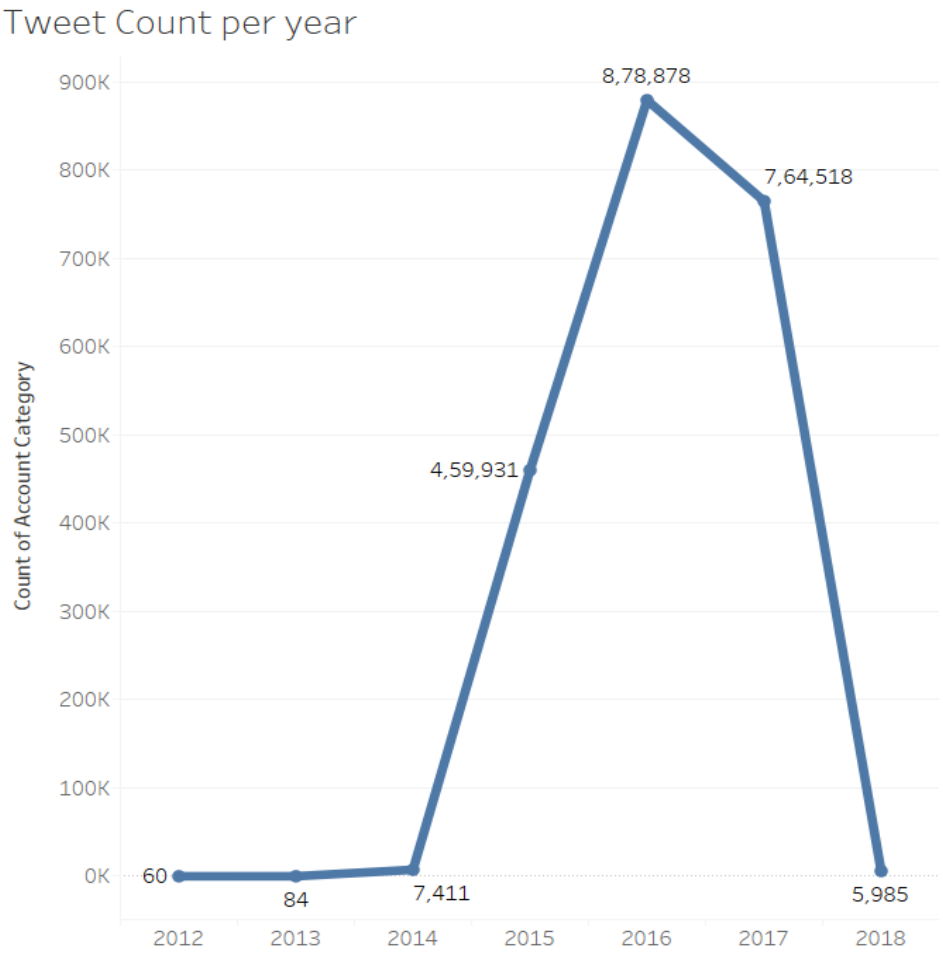

Fig. 11

Now, we have seen the number of tweets published from every account category and the year in which all those tweets were posted. We can observe that most of the tweets were posted from 2015 to 2017. Since, all these years were the campaigning and transitioning years of US presidential elections. Therefore, most of the tweets were in reference with the politics and hence we used our text data modelling concepts like word embeddings to calculate the similarity and biasness in the dataset.

Since, we have seen the account categories from the twitter, now we need to identify the accuracy of this categorizing done by twitter using TF-IDF and similarity concepts of text data modelling and observe how the grouping of tweets is done by content similarity alignment with the labels seen above which is the second question of our assignment.

Grouping Accuracy of Tweets with Account Labels –

In order to find the accuracy of grouping the tweets with the corresponding account labels like Commercial, RightTroll, LeftTroll, etc, we used clustering concept of unsupervised machine learning.

Clustering means segregating data based on the similarity between data instances. It is used to find similarities in the data points and group those data points together to form a cluster. In our dataset, the data points are the tweets posted by all the users and clusters are the account categories. Twitter has tagged each account with a category depending on the tweets they have posted and now we are using clustering technique to see the accuracy of the classification of tweets.

We are using K-Mean Clustering algorithm in our report to identify the similarities and accuracy of the classification. We created clusters from the dataset and placed K centroids in random locations depending on the categories. We assigned all the tweets in the database with cluster labels as per the TF-IDF model we are using. Then we calculated the Euclidean distances of all the data points to each cluster of account category to analyse the closeness of that datapoint to the assigned label.

After calculating all the distances, we created a DataFrame with respect to account categories for observing the number of tweets that got closest distance to each account category cluster. Then, from the original database classification we got the difference in the number of original categorizations of the tweets and the clustering division of our TF-IDF model. With the difference we calculated the % accuracy of categorizing of twitter with respect to our TF-IDF model. In figure 12 below, we can see the parameters and TF-IDF model used for clustering.

```
In [73]: column = 'content'
    ...: TFIDF_PARAMS = {
    ...:     'strip_accents': 'ascii',
    ...:     'stop_words': 'english',
    ...:     'sublinear_tf': True,
    ...:     'ngram_range': (1, 4),
    ...:     'min_df': 0.003,
    ...: }
    ...: vectorizer = TfidfVectorizer(**TFIDF_PARAMS)
    ...: tfidf_model = vectorizer.fit(df[column])
    ...: train_features = tfidf_model.transform(df[column])
```

Fig. 12

In the Table 1 below, we can see the accuracy of grouping of tweets with respect to the content similarity as per the labels assigned by the database.

Account Category	% Accuracy
RightTroll	78
Fearmonger	74
NonEnglish	72
LeftTroll	75
Unknown	70
HashtagGamer	71
NewsFeed	83
Commercial	56

Table No. 1

In figure 13 below, we can see the bar graph pictorial representation of the accuracy of grouping done by the twitter APIs for the mentioned Account category labels.

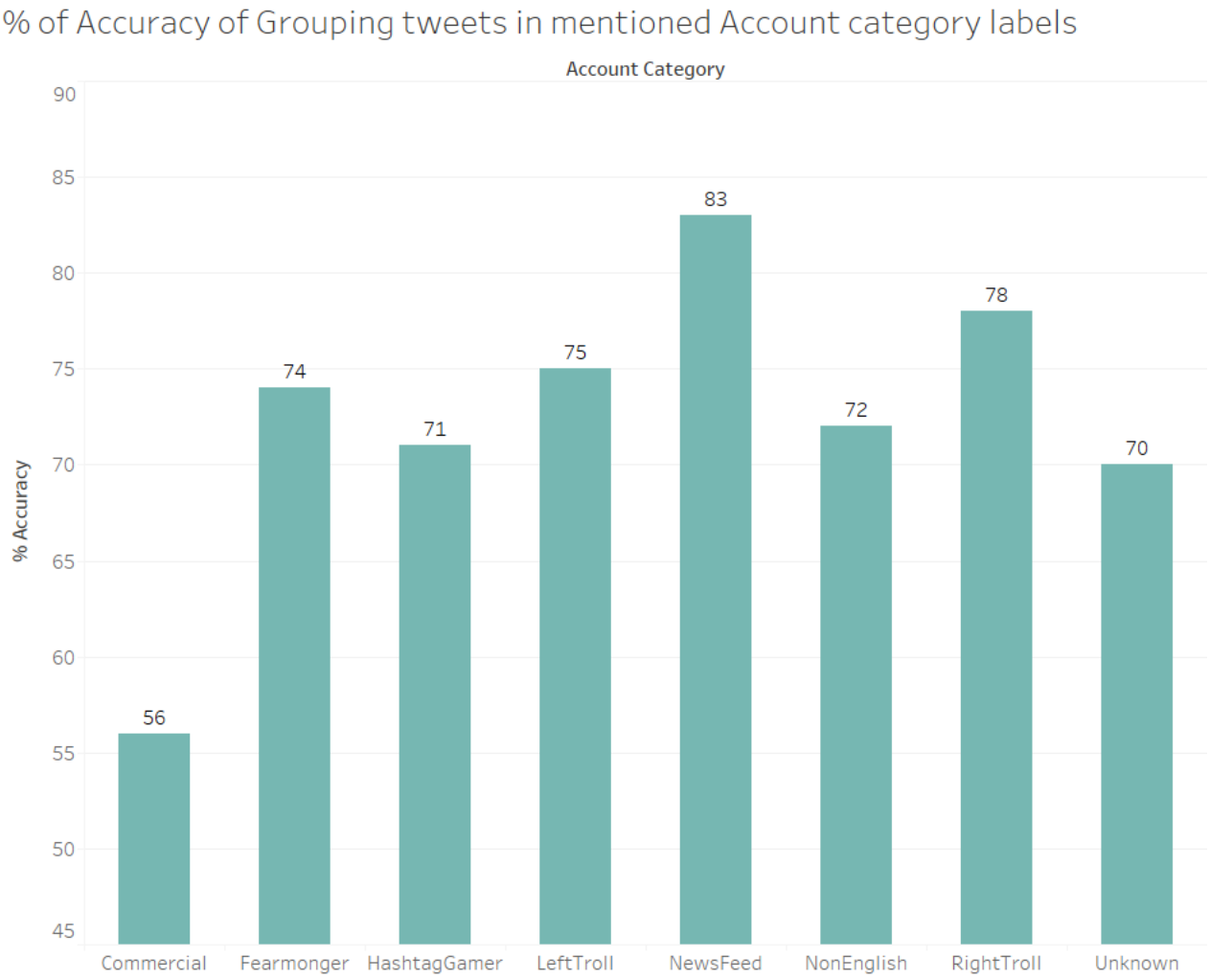

Fig. 13

Conclusion –

We have determined some compelling insights from the dataset given with the help of data modelling concepts like word embeddings, TF-IDF, clustering, relational and temporal modelling. We have completed the requirement of identifying the bias in the data and computing the accuracy in the alignment of tweets with the content similarity. We linked these two queries with our third question of estimating the number of tweets that were posted every year and checking the number of tweets that are categorized in every given label which came to our aid while calculating the accuracy percentage in the grouping requirement. Also, the use of similarity calculation while identifying bias was utilized in accuracy rate calculations as well.

Hence, we have studied and implemented text data modelling concepts to answer two given queries and we have used our third query to link those two queries together with the help of relational and temporal models.

APPENDIX

Reference List –

[1] https://towardsdatascience.com/tf-idf-explained-and-python-sklearn-implementation-b020c5e83275

[2] https://towardsdatascience.com/bias-in-natural-language-processing-nlp-a-dangerous-but-fixable-problem-7d01a12cf0f7

[3] https://towardsdatascience.com/a-beginners-guide-to-word-embedding-with-gensim-word2vec-model-5970fa56cc92

[4] https://towardsdatascience.com/weekly-supervised-learning-getting-started-with-unstructured-data-123354dad7c1

Tools and Modules used –

- Python:
  - Pandas
  - Gensim
  - NLTK
  - MatplotLib
  - RE
  - Contractions
  - NumPy
  - SKLearn

    - IterTools

- Tableau
- MS Word
- Notepad++

Figures and Table descriptions –