Text Analysis using News API

CMPT 732: Programming for Big Data - Project Report

Kanika Sanduja – 301347347
ksanduja@sfu.ca

Supreet Kaur Takkar – 301350081
stakkar@sfu.ca

Vishal Shukla – 301337060
vshukla@sfu.ca

1. Problem definition

Analytics is about obtaining the right information from the huge amounts of available data. With the rapid growth of unstructured data in the past few years, this task has become difficult. Topic Modelling is a statistical method in the field of text mining which can analyze large amounts of unlabeled data and retrieve information by automatically identifying the topics present in a text corpus and deriving hidden patterns. Such models can distinguish between the syntactic differences between the word forms (called inflections) and between multiple meanings of the words and return these abstract topics which can represent the information in the collection. The topics referred here are abstract concepts which consist of a cluster of words that frequently occur together.

News articles found online contain a vast amount of data which can be used to analyze human behavior, inclinations and motivations. News play a significant role in molding public opinion and it lets people have information from around the globe right on their fingertips. It also helps them make well-informed decisions regarding procuring products and availing services from service providers. It seemed interesting to quantitatively analyze and compare the topical coverage of various news sources to see if they can give us an idea about the veracity of the sources.

2. Methodology

We implemented Topic Modelling in our project to analyze the text from the news API: https://newsapi.org/. To implement our idea in the project, we used the following high-level technology stack –

i. Used NewsAPI.org provided API to fetch news headlines and a brief description about the news story and stored them in Cassandra database.
ii. After preprocessing, made use of Latent Direchlet Allocation for topic modelling.
iii. Used Tableau for visualization and to observe interesting patterns from the results obtained.

These are described in detail below.

2.1. Design: Saving Data in Cassandra

We used a Cassandra table (newsdata) to save the data fetched from the News API, for which we used the following schema:

S.No.	Field	SQL Data-Type
1	source	text
2	title	text
3	author	text
4	publishedat	timestamp
5	url	text
6	urltoimage	text

Table 1: Schema of newsdata table

2.2. Data Acquisition: Getting the Data

newsApi provides a very user-friendly API. Upon signing up, we get an API key with which we can make requests for headlines and metadata from various sources. The data acquisition script was written in Python and it inserted the API returned fields into the Cassandra table in batches (script uploaded in repository).

{
Source: techcrunch
Title: Facebook Stories replaces Messenger Day with synced cross-posting
Description: Facebook is cleaning up the redundancy in its Snapchat Stories clones. Today Facebook is killing off the Messenger Day brand and merging the chat app.
Author: Josh Constine
PublishedAt: 2017-11-13T18:00:45Z
URL: https://techcrunch.com/2017/11/13/rip-direct-and-messenger-day/
}

Fig 1: Sample fields returned from the NewsAPI.

2.3. Preprocessing: Prepping it up

The headlines and a brief description about the news story was concatenated into a single text field and it was passed through the following text pre-processing steps:

2.3.1. **Removal of numeric and special characters**

Since numeric and special characters are not required in topic modelling, these were removed.

2.3.2. **Removal of Stop-words**

Stop words are common terms such as *is*, *the*, etc. that are not informative with respect to the content of a document. These need to be removed to make sense of the corpus for topic modelling. Built in stop-words list provided in Python's stop_words package was used in cleaning the text during preprocessing.

2.3.3. **Lemmatization**

For the words *key* and *keys*, we would want to consider these two forms identical as they represent the singular and plural forms of the same word. But we wouldn't want to do the same for the words like *logistic* and *logistics*, which may have different meanings even though they look similar. *Lemmatization* refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the *lemma*.

2.3.4. **Conversion to Lowercase and Tokenization**

Conversion to lowercase is important to consider *keys* and *Keys* as one and *tokenization* is the breaking up text into components. For example: 'Yesterday, I had a fantastic day', when tokenized would generate a list of word: [Yesterday, I, had, a, fantastic, day]

2.3.5. **TF-IDF**

TF-IDF stands for *term frequency-inverse document frequency*, and the TF-IDF weight is a statistical measure used to evaluate how important a word is to a document in a collection of words. This increases with the number of times a word appears in that article while being offset by how frequently that word appears across the entire set of documents. To implement this concept in PySpark, a sequence of SparkML's CountVectorizer and IDF estimators were applied on tokens.

2.4. The LDA Algorithm: How we did it

In Latent Dirichlet Allocation, each news article is viewed as a mixture of topics present in the collection of articles (called the corpus), and each topic as a mixture of words. When a collection of documents is given as an input to the Topic modelling technique, it will identify the word distribution in each topic and topic distribution in each document.

For implementation, we used SparkML's LDA with 'Expectation Maximization' (EM) as the optimizer to take advantage of DistributedLDAModel.
The result obtained from LDA is:

i. Topic distribution for each document/news record
ii. Terms with sorted weights in each topic

2.5. Post-processing: After we were done

To convert the Spark ML LDA output exported to .csv files to simple tables which can then be used for Tableau visualizations, post-processing had to be performed. Date-time inconsistencies were handled, topic wise probabilities for each news article produced and the term weight tables for each topic derived. With the outcome of post-processing script being Tableau Data Source friendly tables.

3. Problems

3.1. **Installing NLTK and its data files on the cluster:** 'pip install' did not work with NLTK on the cluster as installing third-party libraries require more work than that. Also, NLTK data files were downloaded, permissions changed for the home directory so for it be visible to all executors.

3.2. **Exporting the output of LDA to csv files:** This gave error as LDA output contained complex data types in the Dataframe's columns. Conversion to strings before writing to .csv files finally solved the issue.

3.3. **Input to Tableau:** After LDA was implemented, the output of the algorithm had two matrices having document-topic probabilities and topic-word probabilities. The structure of these csv's were not a friendly input to tableau. Post-processing on data was hence performed to make it suitable to be used as a tableau data-source.

4. Project Results

4.1. Outcomes

The category-wise comparison of news sources revealed interesting coverage patterns over time. Some news sources frequently reported a particular topic whereas others covered the topic less often.

4.2. Visualizations

We used Tableau to make time series charts, heat-maps, stacked bar charts, and word clouds. Time series charts were used to compare news sources. Moreover, topics did not come with labels; rather, we had to interpret by ourselves what each topic may mean. We did this by looking at the top 10 words in each topic using word clouds in Tableau.
Click here to see our Tableau Public link for more insights.

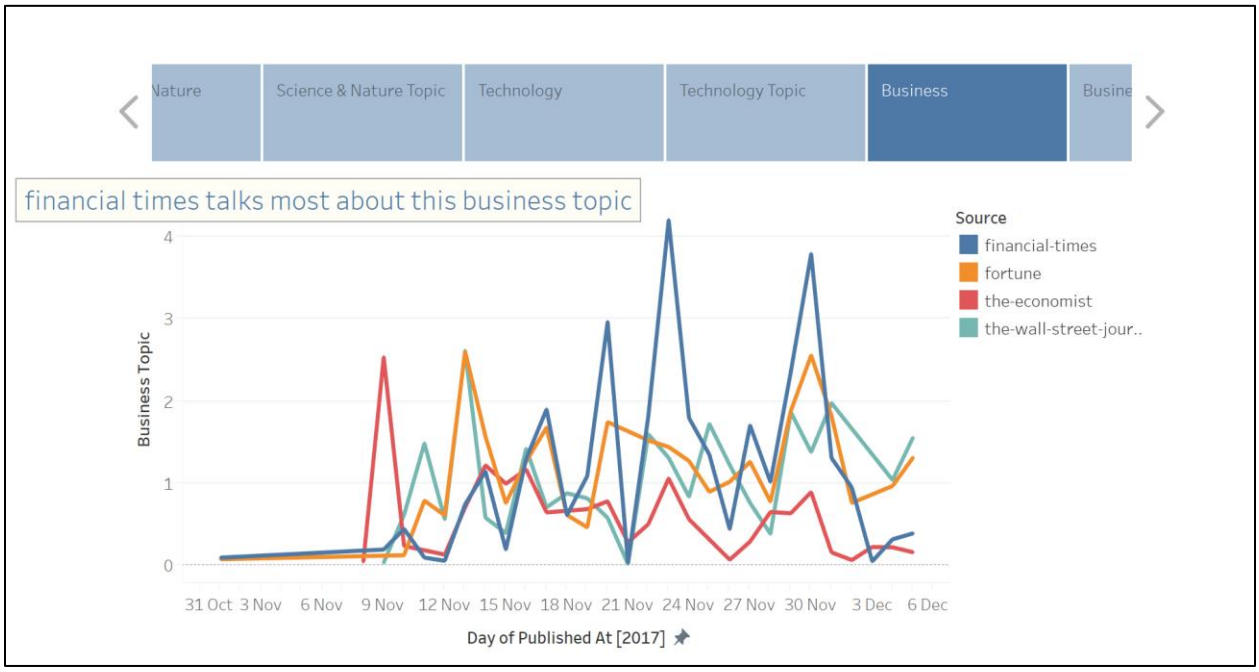

Figure 2: Time Series plot: For the Business category, out of the four sources listed, financial-times talks most about the business topics.

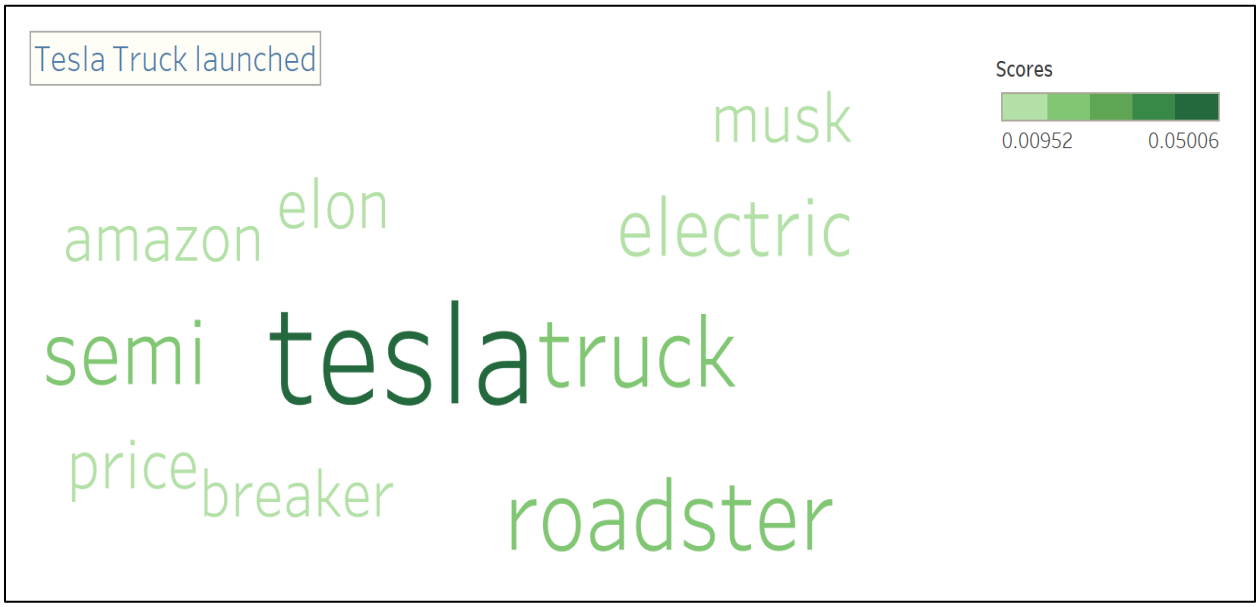

Figure 3: Word-cloud: Reveals the business topic; Darker words are the more probable to appear in this topic

5. Project Summary

5.1. **Data Gathering:** Collected headlines and metadata from 30+ news sources using News API and prepared database for analysis on Apache Cassandra.

5.2. **ETL:** ETL was required right from step one to handle incompatible date-time formats and inconsistent data with missing values. In order to perform text analysis, preprocessing steps were required – like removal of stop words, lemmatization, tokenization, TF-IDF, etc. to clean and normalize text.

5.3. **Problem statement and motivation:** The motivation for our project was to analyzing the topic-wise coverage by various news sources belonging to categories like business, sports, politics, science and nature, technology, etc. We compared various sources for the topics being reported over time using time-series charts and derived probabilities of occurrence of topic in a document.

5.4. **Algorithmic work:** Used Natural Language Processing technique called Latent Dirichlet Allocation to identify abstract topics from the text.

5.5. **Bigness/parallelization:** To keep the implementation distributed and scalable, we used Cassandra for database, and Apache PySpark for text preprocessing and manipulation. SparkML was used to implement topic modelling instead of using other non-parallelizable options to implement in other packages of Python.

5.6. **User Interface:** Tableau was used for presenting interesting analyses and comparisons through time series charts, heat-maps, stacked bar charts and word clouds.

5.7. **Visualization:** Visualization of analyzed results is presented.

5.8. **Technologies:** NLTK library is used for text preprocessing. To better parallelize, we implemented LDA in Spark ML's clustering API instead of other Python package implementations. We used Tableau to present our insights and visualization using new techniques which better fit our outcomes.

Our score to ourselves: 18/20, to always have space for improvement.

6. References and Links

6.1. References

- https://www.tableau.com/beginners-data-visualization
- https://www.tableau.com/learn/training
- https://spark.apache.org/docs/2.2.0/ml-clustering.html#latent-dirichlet-allocation-lda
- https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html
- http://www.tfidf.com/

6.2. Links

- Tableau Public for Text Analysis using News API: https://goo.gl/MiDK7u
- CSIL Repository: https://csil-git1.cs.surrey.sfu.ca/ksanduja/BD-NewsAnalysts.git