Understanding News

Introduction

News is important for a number of reasons within a society. Mainly to inform the public about events that are happening around them and that may affect them.
But **what if these news channels that we religiously follow are biased**, and we only get out half the sentiment? **What if different news articles give out different sentiments and you are left confused at the end?**
And, If we were under the impression that the mainstream media is unbiased and committed to reporting “just the facts,” then the media spectacle in recent days should disabuse us of that outdated notion that the news is objective and unbiased.

Agenda

We are a news outlet catering to people on a monthly basis presenting to them a 30 days trend of a topic with an unbiased view .

We fetch articles from **one or more news sources** and assess the sentiment of each article presenting a **larger picture of the tone of the article** without the need for dwelling deeper into each article for understanding the sentiment.

Problem statement

Use cases

The problem statement and use case that we address in our application is for an end user, who has a list of topics he wants to research about to understand the sentiment pattern and tone of the articles written about that topic.

To perform an analysis for an end user who aims to understand the trending patterns on a variety of topics, we build a model that captures details of an article at a granular level by:

- Accessing a **vast dictionary, that is auto-evolving** (our news articles from the API) and

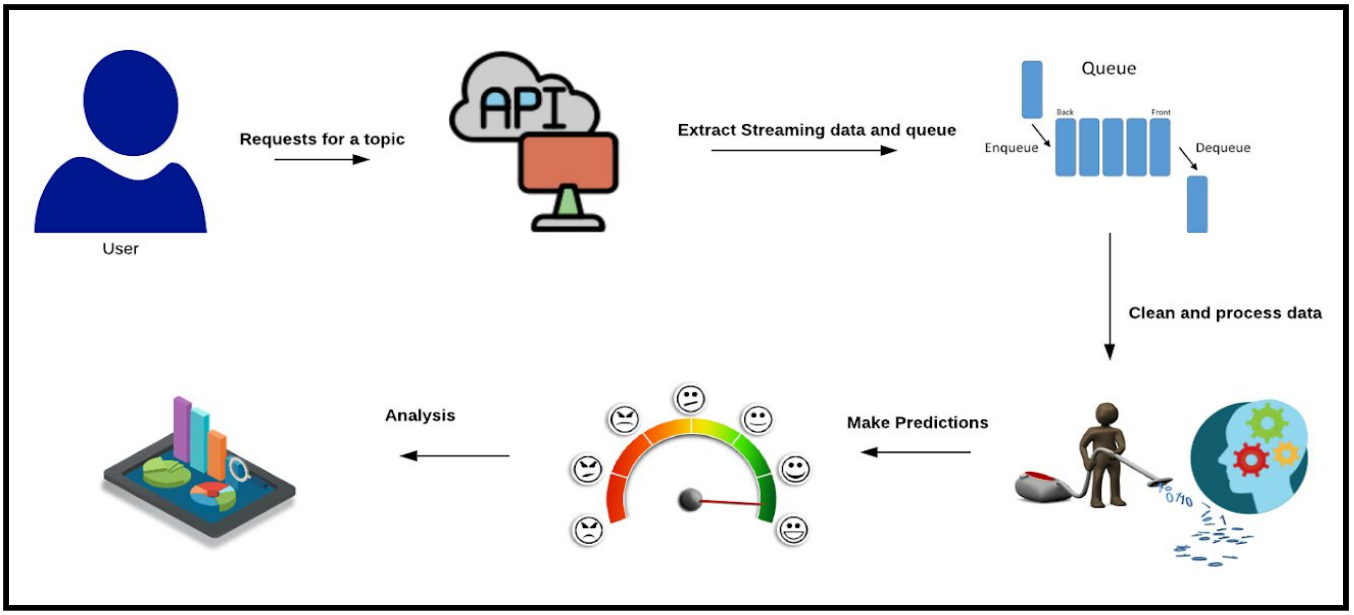
- Giving out the end user a **detailed trend pattern of emotions** and aim to generate a biase-less sentiment score.

Who can benefit:

- **First layer**: Prime user, (eg magazine company owner)

- **Second layer**: Executives of any organization who would want to purchase the service to understand the trends. Eg, stock market traders, data scientists, PR and HR teams of an organization based on the sentiment floating in public about the org, sales and marketing tro understand projections, political leaders on how their image stands in

Data flow:



Step 1

User searching for a specific topic and specify the timeframe withing which the results would be presented

Step 2

Fetching all data within the defined timeframe

Step 3

Extracting the streams of data and queue them using a pub/sub environment.

Step 4

Cleaning and provessing the data.

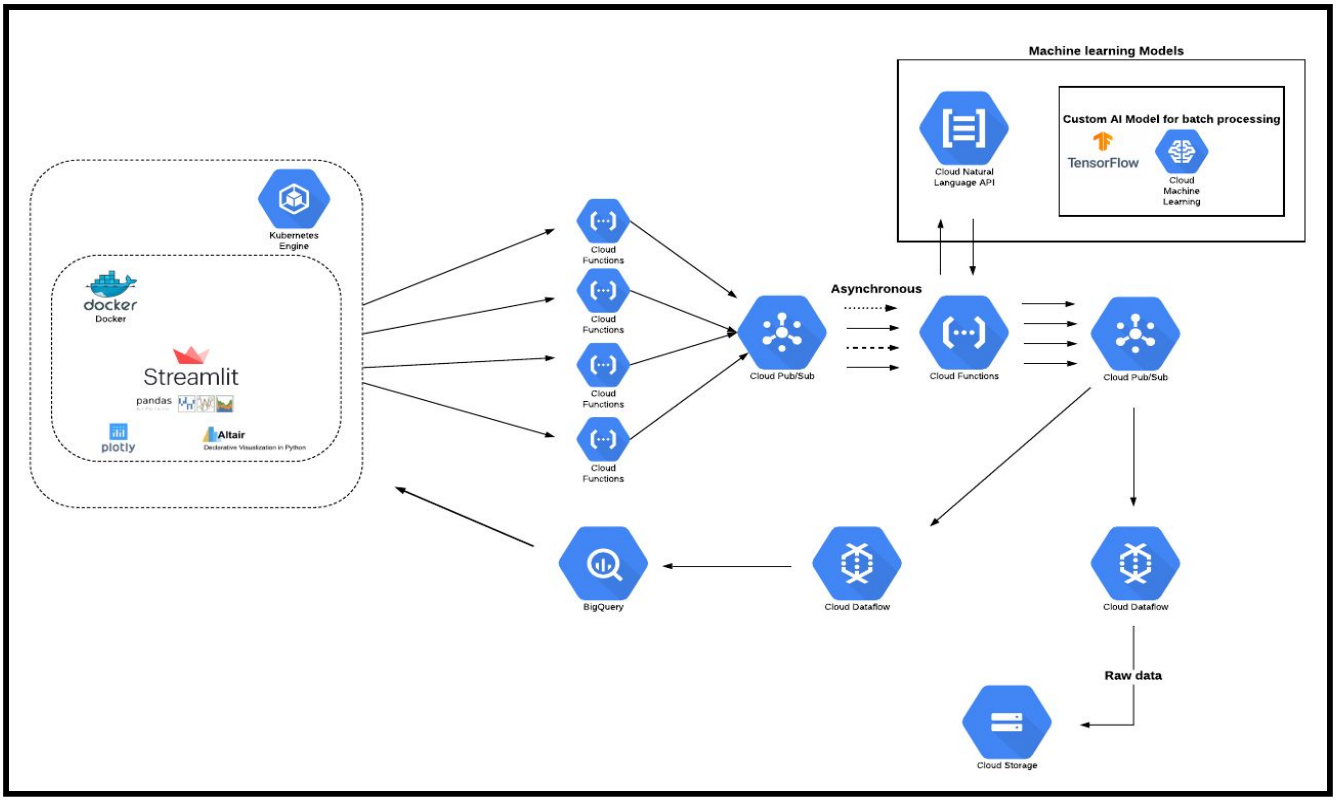
Step 5

Sending the cleaned data to a machine learning model for predictions.

Step 6

Analysing the data and presenting the final analysis

Architecture and Implementation



Streamlit:

We would be using Streamlit as an interface between the user and our pipeline. The user would search for any topic which would instantiate our pipeline for processing the topic and plot back the results of the search.

Cloud Function 1

Google Cloud Functions is a serverless execution environment for building and connecting cloud services.
In our case, this cloud function is used to make requests to various news sources. The search request is based on the topic searched by the user. The api requests all the news articles which are even remotely related to this topic and returns back streams of articles.

Cloud Pub/Sub 1

Cloud Pub/Sub is a global message queue that is part of the Google Cloud Platform. It works on the fundamental messaging concepts like Queues, Topics, Messages, Subscribers and Subscriptions.

Since the data we are fetching using our cloud functions is streaming data, we would need to queue this data to be able to process it without losing any of the data.

Cloud Function 2

Publish/subscribe messaging, or pub/sub messaging, is a form of asynchronous service-to-service communication used in serverless and microservices architectures.

Here, the pub/sub makes multiple asynchronous calls to the cloud functions which triggers our machine learning module to analyse the sentiment of each article.

Machine Learning Models

We have two machine learning models through which the api calls can be made.

The first model is a **Google Natural language processing model** has has been pretrained by google and provides an API to call the model. The second model is **our custom sentiment analysis model** that has been **trained by us and deployed on Google Cloud Platform under custom models module on the google AI Hub**.

We would be using the **Google Natural language processing model** for our streming data predictions and would opt for our Custom model while making predictions for batch data.

Sentiment Analysis inspects the given text and identifies the prevailing emotional opinion within the text, especially to determine a writer's attitude as positive, negative, or neutral. Sentiment analysis is performed through the **analyzeSentiment** method.

Cloud Pub/Sub 2

Since the cloud function makes anynchronous requests to the ML API, the results are received in a similar asynchronous format. Thus to avoid the conjestion, we are using the second Pub/Sub queue.

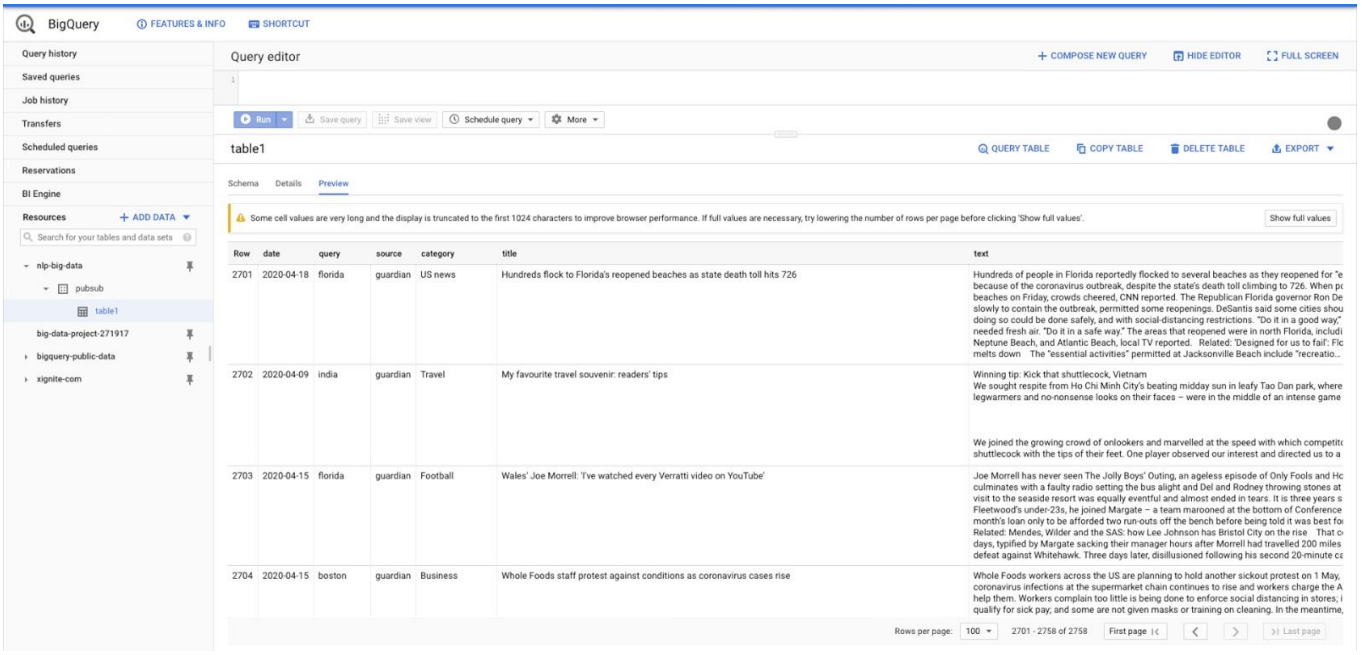
DataFlow

We stream the data again to the two dataflow functions, one to save the **raw data as checkpoints in the bucket** and the second dataflow is used to send the **stream this data to BigQuery**

BigQuery

BigQuery is a serverless data warehouse that allows storing data (up to Terabytes) and runs fast SQL queries without worrying about the computing power.

We would be storing the data and processing this using the SQL queries in bigtable to fetch the requested analytics by the user



Result Analysis

When we read multiple articles written about the same topic, we are left with a diffrent sentiments at the end of each read. How can we summarize the sentiment and understnd the overall emotion towards that topic.

Interpreting sentiment analysis values

The **score of a document's sentiment** indicates the **overall emotion** of a document. The **magnitude** of a document's sentiment indicates **how much emotional content is present within the document**, and this value is often proportional to the length of the document.

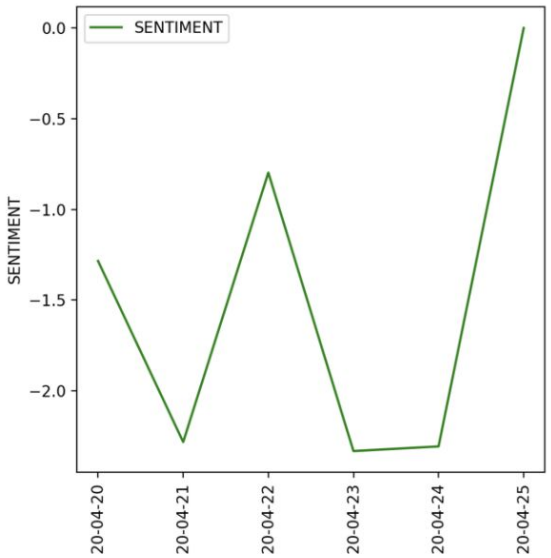
It is important to note that the Natural Language indicates differences between positive and negative emotion in a document, but does not identify specific positive and negative emotions. For example, "angry" and "sad" are both considered negative emotions.

A document with a **neutral score (around 0.0) may indicate a low-emotion document**, or may indicate **mixed emotions,** with both **high positive and negative values which cancel each out**.

We use **magnitude values to disambiguate these cases**, as truly neutral documents will have a low magnitude value, while mixed documents will have higher magnitude values.

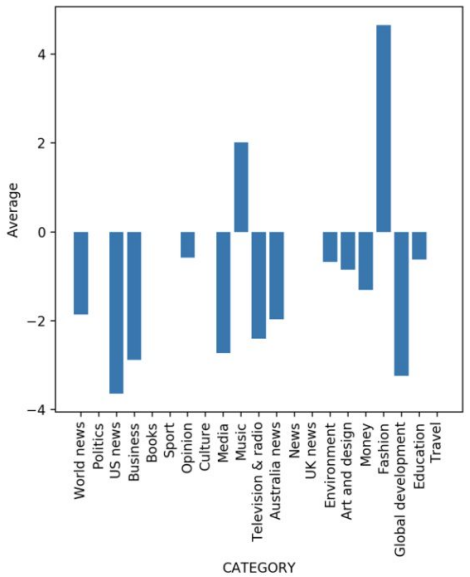
Our primary focus is to present the user with an overall sentiment about a specific topic from various news articles and sources over a period of time.

1. Overall Sentiment over a period of time



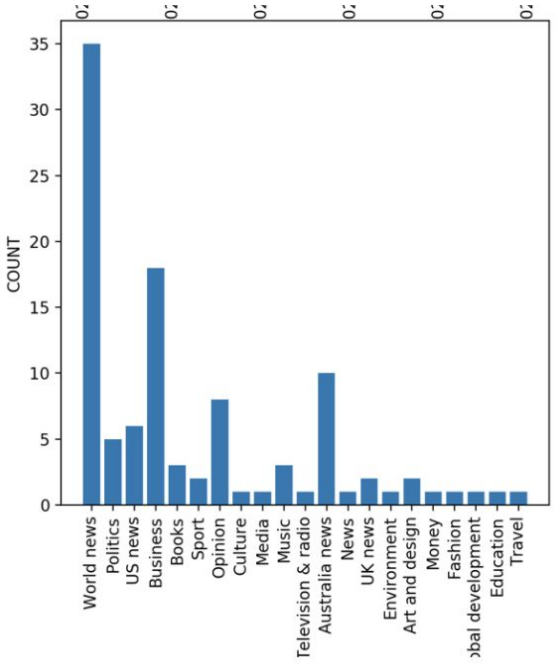
Secondly, we wish to understand the different categories associated with this topic. This gives us an understanding of the association. We analyze how many articles in a category have the mention of the topic.

2. Number of articles in each category



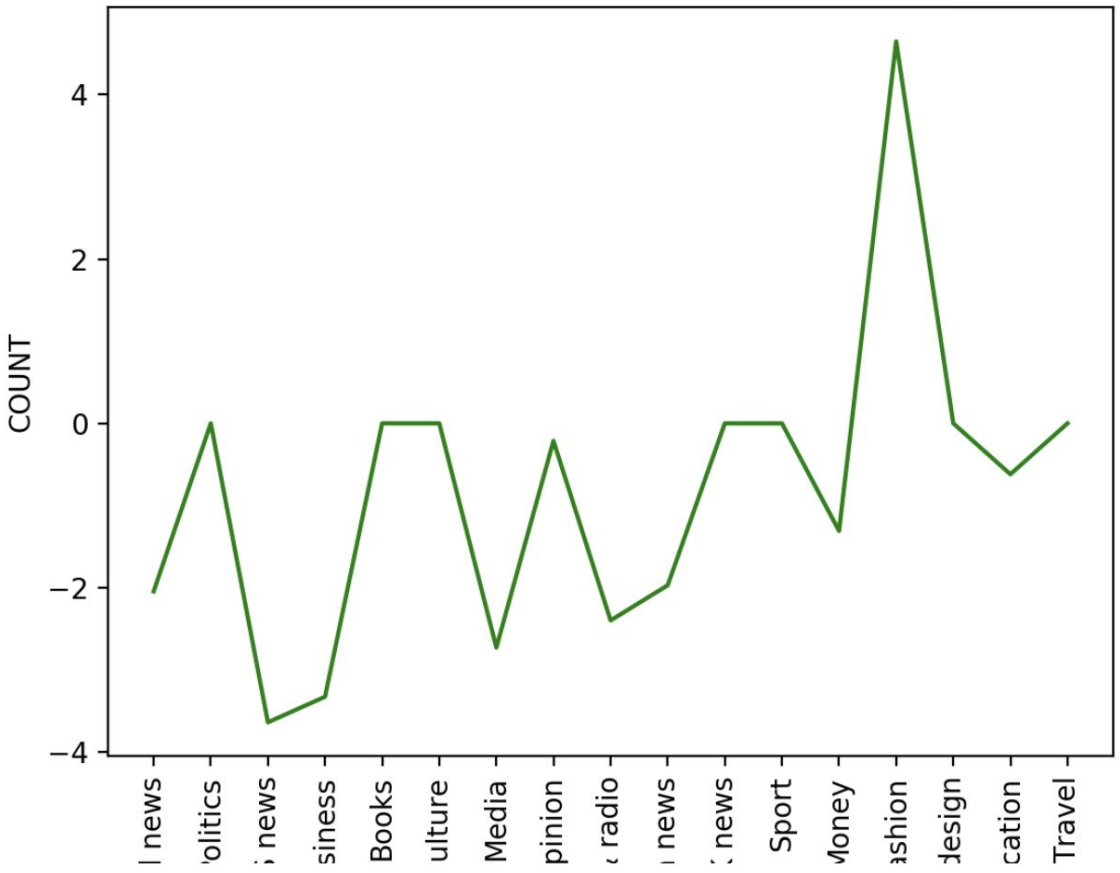
Then we analyze category-wise sentiment analysis. This helps the user understand what is the sentiment of the topic in a specific domain (eg. is the sentiment of oil positive or negative and by how much in the business domain)

3. Sentiment in each category



We also analyze the cross correlation between two topics. For instance, Donald Trump and Jobs. This gives us a better idea of how the two topics influence each other.

4. Cross correlation with the topic



Test cases

Factors considered while preparing test cases:

1.Positive: The test cases should include the positive as well as the negative test case.Testing the application with a positive test case means To test the field with the value which actually the field needed.

2.Negative: While we perform testing the negative test cases play a very important role.We assume that the application will run with all the needed values entered but how the system reacts when an invalid value is entered is important too.
So negative test cases are very important.

3.Usability: Testing the efficiency and accuracy of the application.Whether the user is able to perform the task successfully or not.

4.Performance: Test cases should include the conditions for the testing workload.

5. Reliability: It means testing the consistency of application by performing the same task repeatedly.

Streamlit User Interface:

Project Name:	Serverless NLP - Understanding News						
Module Name:	Streamlit User interface						
Created By:	Manogna						
Test Case Category	**Test Case description**	**Prerequisite setup steps(if any)**	**Data required**	**Test Step**	**Expected Result**	**Actual Result**	**Status**
Positive test	1. Input should be a string (alpha numeric is accepted eg. WorldWar 2) 2. Data range should be selected 3. Cross topic is optional	Build stremlit and run on localhost server	News articles fetched from API	1. Input topic as 'boston', 2. datarange select radio button for 5 days, 3. cross topic lefty blank	Retrn results with 3 plots	Returned results with 3 plots	Pass
Negative Test	1. Does not accept empty value in topic name 2. Enter non existent words	Build stremlit and run on localhost server	News articles fetched from API	No input	1. Throws warning as no input is passed 2. Returns results as "no results found" since the word does not exist	1. Throws warning as no input is passed 2. Returns results as "no results found" since the word does not exist	Pass
Usability	Receiving meaningful and relevant results	Build stremlit and run on localhost server	News articles fetched from API	Run query	Relevant and consistent results	Relevant and consistent results	Pass
Performance	1. Results are faster on prerun queries 2. Results are faster on queries with shorter date range	Build stremlit and run on localhost server	News articles fetched from API	1. Run query on same data again 2. Select new query for shorter	Return results faster	Returned results faster	Pass
Reliability	Reproduce same result with same query	Build stremlit and run on localhost server	News articles fetched from API	Passing same queries as above	Returns the same plots	Returned the same plots	Pass

Cloud Functions and Pub/Sub

Project Name:	Serverless NLP - Understanding News
Module Name:	Cloud Functions and pub/sub
Created By:	Manogna

Test Case Category	Test Case description	Prerequisite setup steps(if any)	Data required	Test Step	Expected Result	Actual Result	Status
Positive test	1. Cloud Function1- Fetching articles from News API : returns only article	Create cloud function and set up topics for Pub/Sub	topic name from user(frontend)	Running cloud function 1	Returns the article with date, source, category, title and text	Returns the article with date, source, category, title and text	Pass
Negative Test	Cloud function 2: Images and text in data	Create cloud function and set up topics for Pub/Sub	text(as input)	Instantiating the cloud function 2	Captures only text and leaves out the images and html, css tags	Captures only text and leaves out the images and html, css tags	Pass
Usability	Receiving the sentiment and magnitude	Enable and setup Cloud NLP API	text(as input)	calling NLP Api	Returns sentiment value between -1 and 1 and magnitude for each article	Returns sentiment value between -1 and 1 and magnitude for each article	Pass

Future scope

In the future, we would like to extend our model to perform text summarization, entity sentiment analysis not only limiting it to the News articles but also to twitter data.

References

- https://imadelhanafi.com/posts/bigquery_dashboard/
- https://www.oodlestechnologies.com/blogs/The-importance-of-Test-Case-In-Software-Testing/
- https://towardsdatascience.com/game-of-thrones-twitter-sentiment-with-keras-apache-beam-bigquery-and-pubsub-382a770f6583
- https://www.edureka.co/blog/test-case-in-software-testing/