Course Assignment

CZ4034 Information Retrieval

Names: Brendan Peng Chuan Hyde U1722723L

Lee Qian Yu U1720227F

Lin HuangJiayin U1721208D

Okkar Min U1722504A

Tammy Lim Lee Xin U1822254G

Tay Jaslyn U1721893B

Date: 9th April 2021

Abstract

In this report, we discuss on how our group built an information retrieval system from ground up that enables user of the system to understand sentiment of a selected market.

External Links

Youtube:https://youtu.be/e8FTQv6DAys

Gdrive:https://drive.google.com/drive/folders/1IlUWdSs0cNKG4M35Z_92UMLcsMcJVMKx?usp=sharing

1. Business Question

Our group would like to answer the following question:

- ‘What are my consumers talking about my product?’
- ‘Are my consumers generally positive/neutral/negative?’
- ‘Which is/are the products of my company that can be improved upon?’

By having answers to the questions, businesses decision makers would be able to improve consumer/customer satisfaction.

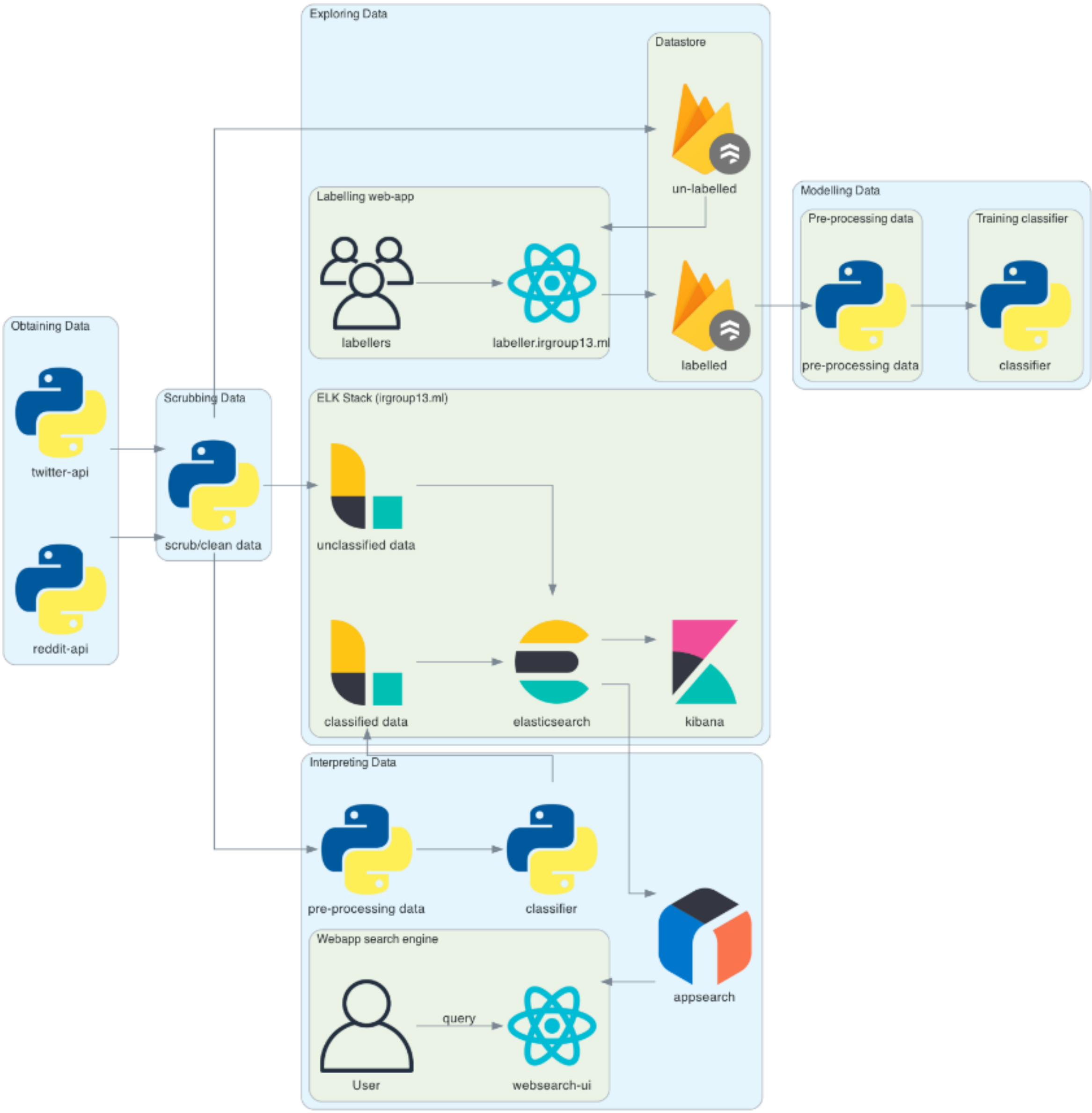
2. Companies

As there are many varieties of businesses, our group mainly focus on 10 technical companies that are publicly listed.

1. Adobe
2. Apple
3. Amazon
4. Facebook
5. Google
6. Microsoft
7. Nvidia
8. Salesforce
9. Samsung
10. Tencent

3. Methodology

We adopt the **OSEMN** framework, which is the five stages lifecycle of a data science project. We will dive deep into details regarding each stage below. See image below for a better overview on how our Information Retrieval System is setup.



Information Retrieval System

1. **O**btaining Data
2. **S**crubbing Data
3. **E**xploring Data
4. **M**odelling Data
5. I**n**terpreting Data

3.1 Obtaining Data

Data is obtained from Twitter [1] and Reddit [2] API that enable us to get user posted content. Data is obtained daily using scheduled python cron job at 12AM GMT+8. Obtained data is stored in JSON format in following shape, we call this a document:

```
{
    "dataID": 1,
    "body": "The quick brown fox jumped over the lazy dog",
    "company": "exampleCompany",
    "created_utc": 123456789,
    "score": 5,
}
```

Key	Description
dataID	Unique ID to keep track of
body	Textual data of tweet/reddit comment
company	One of 10 companies listed above in **2. Companies**
created_utc	Coordinated Universal Time that tweet/reddit comment was made
score	Quality Score of tweet/reddit comment
data_from	Source of the crawled data. (Reddit or Twitter)

For twitter data-crawl, we found that the usage of mentions (@) was more appropriate than hashtags. The users tend to be saying something directed at the brand/company when mentions were used, whereas hashtags were used to drum up the popularity of the tweet. We summed up the number of favourites, retweets, and replies and used that as a quality score for the tweet.

The official Twitter API does limit the number of tweets we may crawl. For example, as we used the recent search API to crawl the data from the last seven days, this adds to the monthly cap of 500,000 tweets. The alternative to this would be running a stream API to capture the live tweets as they are made. However, this was not necessary as the data need not be live, and we would be able to get about 10,0000 tweets per week through our recent search API, that more than sufficed.

To crawl the Reddit corpus, PRAW (A Python wrapper for Reddit API) was used to authenticate and use the Reddit API functions. The Reddit API has a rate limit of up to 60 requests per minute and has a limit of 100 items per API call. As such, since our requests require 10 subreddits * 100 items = 10,000 items, PRAW breaks it into multiple API calls with a slight delay.

For each Reddit comment, we retrieve three meta information namely: *body, score* and *created_utc*. The *body* retrieves the comment posted, the *score* retrieves the "karma" which represents the number of upvotes - number of downvotes a comment receives and the *created_utc* retrieves the timestamp in which the comment was being posted.

3.2 Scrubbing Data

Before the raw data is pushed to our database, we performed a rudimentary clean up. This involved removing the mentions, twitter links, as well as restructuring the data to fit the standard. We also realized that emojis are commonly used in both Twitter and Reddit which could be meaningful. Hence, we used a python library to translate the emoji into their relevant text descriptions.


```
{"text": "@Adobe thank you for updating Adobe illustrator for the iPad. \n\n\ud83d\ude4f\ud83c\udffe", "author_id": "2750941496", "public_metrics": {"retweet_count": 0, "reply_count": 0, "like_count": 0, "quote_count": 0}, "id": "1366005926684147714", "created_at": "2021-02-28T12:42:46.000Z"}
```


Raw twitter data that was crawled from the Twitter API.


```
{"body": "thank you for updating Adobe illustrator for the iPad. folded_hands_medium-dark_skin_tone", "score": 0, "created_utc": 1614487366, "company": "adobe", "data_from":"twitter"}
```


The cleaned Twitter data that was pushed to the database.

3.3 Exploring Data

Data exploration was done by enlisting the help of 2 open-sourced tools and 1 free-to-use database.

1. Next.js (labeller.irgroup13.ml)

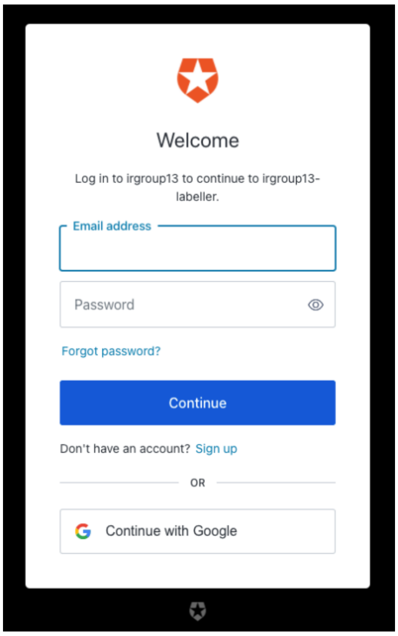
2. ELK Stack (Elasticsearch Logstash Kibana) (irgroup13.ml)
3. Firebase Cloud Firestore

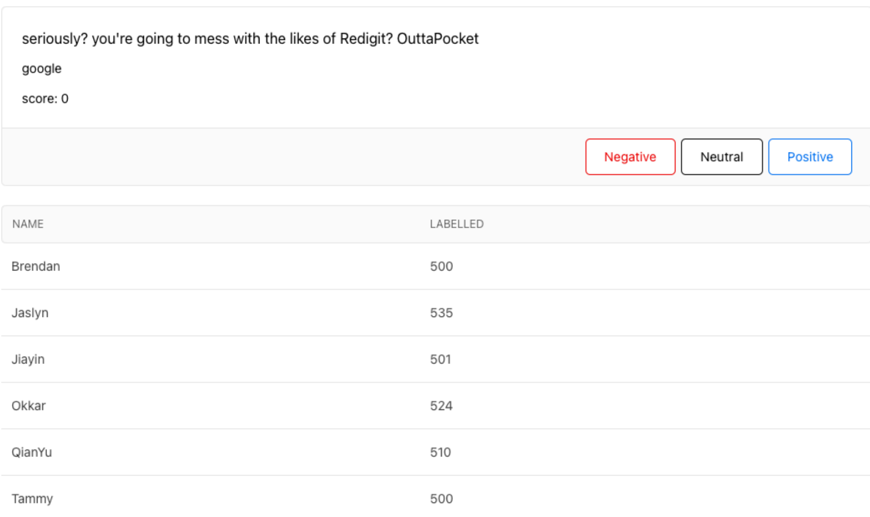
3.3.1 Next.js

Next.js is an open-source React front-end development web framework that enables functionality such as server-side rendering and generating static websites for React based web applications.

We built it to help us with manual labelling of data, to obtain inter-annotator agreement rate and get ground-truth for sentiment classifier training. The inter-annotator pairings are:

- Brendan – Jiayin
- Jaslyn – Okkar
- Tammy – Qianyu





NAME	LABELLED
Brendan	500
Jaslyn	535
Jiayin	501
Okkar	524
QianYu	510
Tammy	500

The web application have 3 simple buttons ‘Negative’, ‘Neutral’ and ‘Positive’. Operation that happens in labelling web-app are as follows:

1. Labeller sign in using Auth0
2. Labeller read the body of text
3. Labeller determine one of ‘Negative’ ‘Neutral’ or ‘Positive’
4. Labeller press the corresponding button
5. Labelled data is saved to firebase (see **3.1 Obtaining Data**)
6. New data is fetched and repeat from 2

Advantage of having a web-app to label is that labellers can label at their own convenience and having leader board of labelled count keep us accountable.

3.3.2 ELK Stack (Elasticsearch Logstash Kibana)

Elasticsearch Logstash Kibana (ELK) is a set of open-source tools that are commonly deployed together to achieve one purpose. To index document. User can define how document are to be indexed. ELK is an alternative to Solr Lucene.

We will discuss about tweaks and improvements we made to indexing and ranking so that user would be able to have better query result in **3.5 Interpreting Data**

3.3.3 Firebase Cloud Firestore

Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud.

We are making use of Cloud Firestore for its real-time ability to update the labelled count on the labelling leader board. It is also notable that Firestore’s Collection Document model fit nicely with our information retrieval system:

- Document: unit of storage is the document. A document is a lightweight record that contains fields, which map to values. Each document is identified by a name.
- Collection: documents live in collections, which are simply containers for documents. For example, you could have a user collection to contain your various users, each represented by a document.

Each tweet/reddit comment is a document which lives in our collection named ‘unclassified’ and ‘classified’.

3.4 Modelling Data

3.4.1 Pre-processing of data for model training

Upon completion of data crawling, around 4,500 records were selected randomly from the collected data. Then, manual labelling of data was conducted in pairs to ensure inter-annotator agreement of at least 80%, as explained in the section above. Each pair was assigned 1,500 records to classify, and only data that were classified as the same class by both members were kept. Unfortunately, due to the relatively low number in positive feedbacks posted by users, it was difficult to collect equal number of records from each class. After this step, data pre-processing is conducted.

Data pre-processing is one of the key steps for natural language processing and it can directly impact the outcome of the final tokens present before undergoing machine learning and classification. For data collected from social media, it is common that it is informal, contain slangs or wrongly written by author. Therefore, it is necessary to conduct pre-processing to ensure that such data are cleaned with unnecessary characters removed.

Some methods of data pre-processing include noise removal, spelling correction and lemmatization. For this project, pre-processing was conducted in this order:

1. Removing all extra whitespaces and lowercasing alphabets
2. Converting all accented characters to ASCII characters
3. Expanding any contractions
4. Removing any special characters
5. Converting words to numerical format
6. Spell correction for misspelled words
7. Removing stop words and lemmatisation of tokens

Step 1 and 2 comprises of basic noise removal techniques and can help in the reduction in vector matrix. Step 3 is conducted before step 4 as the removal of special characters will cause the apostrophe (‘) to be removed, hence impacting the expansion of contractions. To ensure that the dataset is standardised, conversion of words to numerical is conducted.

Since it is common that users misspell words, step 6 replaces any words that are not found in the dictionary with a word that has the lowest edit distance from the original. In step 7, stop words are removed since they are words with high frequency, and do not bring any value to semantic analysis. As seen in the word cloud under **4.0 Answering Questions in Assignment.pdf** below, the words “one”, “will”, “then” appears frequently in our dataset.

These are examples of stop words that were removed in this process. Finally, the remaining text is lemmatised to ensure consistency and reduce vector space. This results in the first set of pre-processed data.

For the second set of data, lemmatisation was conducted in detail based on its word form, whether it is a noun, verb, or adjective. Spell correction was also removed in considerations that useful slangs that were not included in the dictionary may be wrongly edited and may affect final results. With this, two different pre-processed datasets were formed and used in the following classifier section. An example of a pre-processed data will look like this:

```
Before: when will we have a fix for the disappearing brush
outline on Photoshop???
After: fix disappearing brush outline photoshop
```

3.4.2 Building tri label classifier

Two methods of feature extraction have been adapted on the pre-processed data in the section above. The methods are as below:

1. Count vectorizer (CV): To tokenize and count the occurrence.
2. Term-Frequency Inverse Document-Frequency (TDIDF) term weighting: To calculate the term-frequency (TF) and inverse document-frequency (IDF).

Under count vectorizer, the data strings are tokenized by extracting words with at least 2 letters. The terms are then allocated to a unique integer index that corresponds to the column in the resulting matrix. As such, words that are not in the training corpus will be ignored in future calls. Under TDIDF term weighting, words with less relevance are removed with the re-weighting of the count features. This done by multiplying the TF, the number of occurrences of a term in a given document, with the IDF. The class labels are transformed to one hot vectors.

The following classifications methods have been used:

1. 4-layer long short-term memory (LSTM)
2. 4-layer convolutional neural network (CNN)
3. Support vector machine (SVM)
4. Logistic regression (LR)
5. Random forest (RF)
6. K-nearest neighbours (KNN)

7. Decision tress (DT)
8. AdaBoost (AB)

The approaches chosen are the popular classifiers for sentiment analysis. The model and pre-processing that provides the highest accuracy is then chosen.

Due to the low accuracy of the LSTM, with validation accuracy of 30.47%, 26.20%, and CNN, with validation accuracy of 67.08%, no further exploration has been conducted on them. The classifiers from part 2 to 8 are the tested with the different methods of feature extraction. In addition to the pre-processed data and models, over sampling has been tested. For the oversampling process, the number of positive and negative data have been increased to the amount of the neutral data.

The evaluation metrics used are precision, recall and F-measure. However, only the average F1-score is used for the initial classifiers. With the initial results, the classifiers will then be ensembled together.

Dataset1

Model	CV Accuracy (%)	Oversample + CV Accuracy (%)	TDIDF Accuracy (%)	Oversample + TDIDF Accuracy (%)
SVM	69	87	72	82
LR	72	88	72	83
RF	71	89	70	89
KNN	63	77	62	77
DT	63	86	62	84
AB	68	68	65	64

Dataset2

Model	CV Accuracy (%)	Oversample + CV Accuracy (%)	TDIDF Accuracy (%)	Oversample + TDIDF Accuracy (%)
SVM	68	87	68	82
LR	70	88	69	82
RF	69	89	68	88
KNN	63	79	62	87
DT	66	85	63	82
AB	66	68	66	67

The table shows the following information:

1. There are marginal differences in accuracy between the different datasets, dataset1 and dataset2.
2. There are marginal differences in accuracy between the different feature extraction methods, count vectorizer and TFIDF.
3. There is an increase in accuracy with oversampling, except for AB.

To enhance the classification, stacked ensemble has been adapted. For each feature extraction method and dataset, the models with the top 2 accuracy have been stacked together. Due to a current bug in the existing framework's processor, only 2 models will be combined. Since oversampling of the data provides the greater accuracy, the oversampling data is used for the training of the new model.

		Training Model		Evaluation Model	
Model Combination	**Accuracy (%)**	**Training Time (s)**	**Prediction Time (s)**	**Training Time (s)**	**Prediction Time (s)**
Dataset1 + CV + (LR + RF)	90	29.18	0.14	7.13	0.09
Dataset1 + TDIDF + (SVM + RF)	91	12.52	0.13	7.16	0.09
Dataset1 + TDIDF + (RF)	89	2.53	0.13	1.36	0.09
Dataset2 + CV + (DT + RF)	91	32.71	0.16	7.12	0.09
Dataset2 + TDIDF + (DT + RF)	90	12.63	0.14	7.46	0.09

Using: **Dataset1 + CV + (LR + RF)**

```
Dataset1 + Count Vectorizer + Stack ensemble (LR + RF)
              precision    recall  f1-score   support

    negative       0.88      0.88      0.88       583
     neutral       0.85      0.86      0.85       568
    positive       0.98      0.96      0.97       581

    accuracy                           0.90      1732
   macro avg       0.90      0.90      0.90      1732
weighted avg       0.90      0.90      0.90      1732
```

Using: **Dataset1 + TDIDF + (SVM + RF)**

Dataset1 + TDIDF + Stack ensemble(SVM + RF)

	precision	recall	f1-score	support
negative	0.91	0.87	0.89	609
neutral	0.83	0.90	0.86	517
positive	1.00	0.97	0.98	606
accuracy			0.91	1732
macro avg	0.91	0.91	0.91	1732
weighted avg	0.91	0.91	0.91	1732

Using: **Dataset1 + TDIDF + LR**

Dataset1 + TDIDF + LR

	precision	recall	f1-score	support
negative	0.49	0.68	0.57	205
neutral	0.89	0.72	0.80	727
positive	0.10	0.70	0.18	10
accuracy			0.72	942
macro avg	0.50	0.70	0.52	942
weighted avg	0.80	0.72	0.74	942

Using: **Dataset2 + CV + (LR + RF)**

Dataset2 + Count Vectorizer + Stack ensemble (LR + RF)

	precision	recall	f1-score	support
negative	0.88	0.88	0.88	571
neutral	0.85	0.87	0.86	580
positive	0.99	0.96	0.98	583
accuracy			0.90	1734
macro avg	0.91	0.90	0.91	1734
weighted avg	0.91	0.90	0.91	1734

Using: **Dataset2 + TDIDF + Oversample + (DT + KN)**

Dataset2 + TDIDF + Stacked ensemble(DT + KNN)

	precision	recall	f1-score	support
negative	0.90	0.80	0.85	656
neutral	0.74	0.87	0.80	516
positive	0.98	0.95	0.97	562
accuracy			0.87	1734
macro avg	0.87	0.87	0.87	1734
weighted avg	0.88	0.87	0.87	1734

From the results above, we can see that precision level ranges around 70 to 100, with the mean of 89.73. The recall level ranges around 85 to 100, with the mean of 89.68. The f1-score ranges around 80 to 100, with the mean of 89.6.

A majority vote is then taken from the models above. In order to have an odd number for majority vote, an additional model has been added in. The model chosen has the highest accuracy from the oversampling pool.

	Time taken to predict csv file with 31k data (s)	**Time taken to predict single sentence (s)**
Member 1	79.60	63.31
Member 2	124.45	119.00
Average	102	91

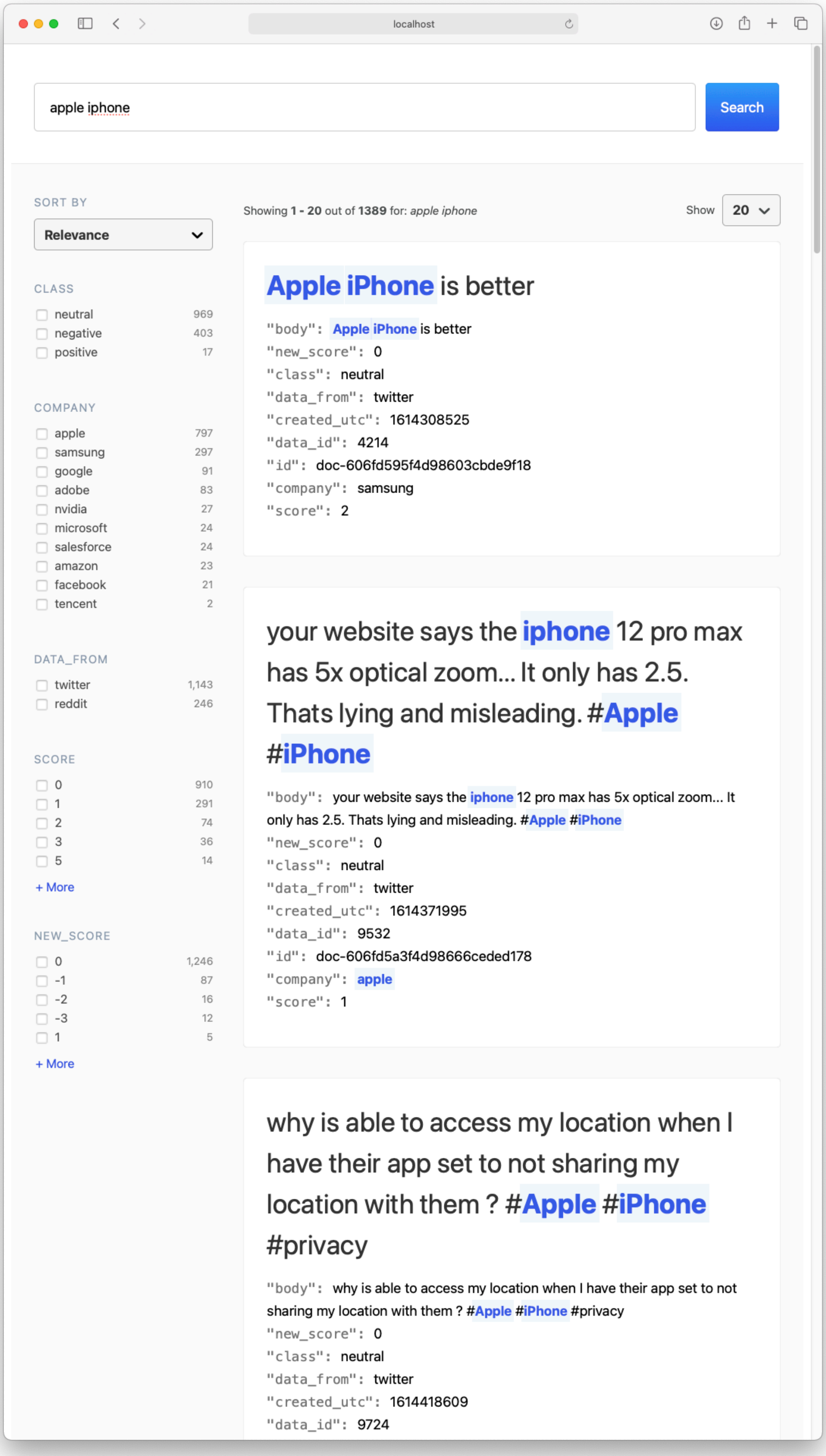
In terms of time, the initial plan for prediction is under 2 minutes. The results provided is an average of 102 seconds for the CSV file, and 91 seconds for a string, both with better timing than our initial plan. Despite saying that, the models can be improved by increasing the ensemble classifications. However, the ensemble is currently restricted by a bug in the framework. Alternative, the data could be processed in a way that allows more than 2 models.

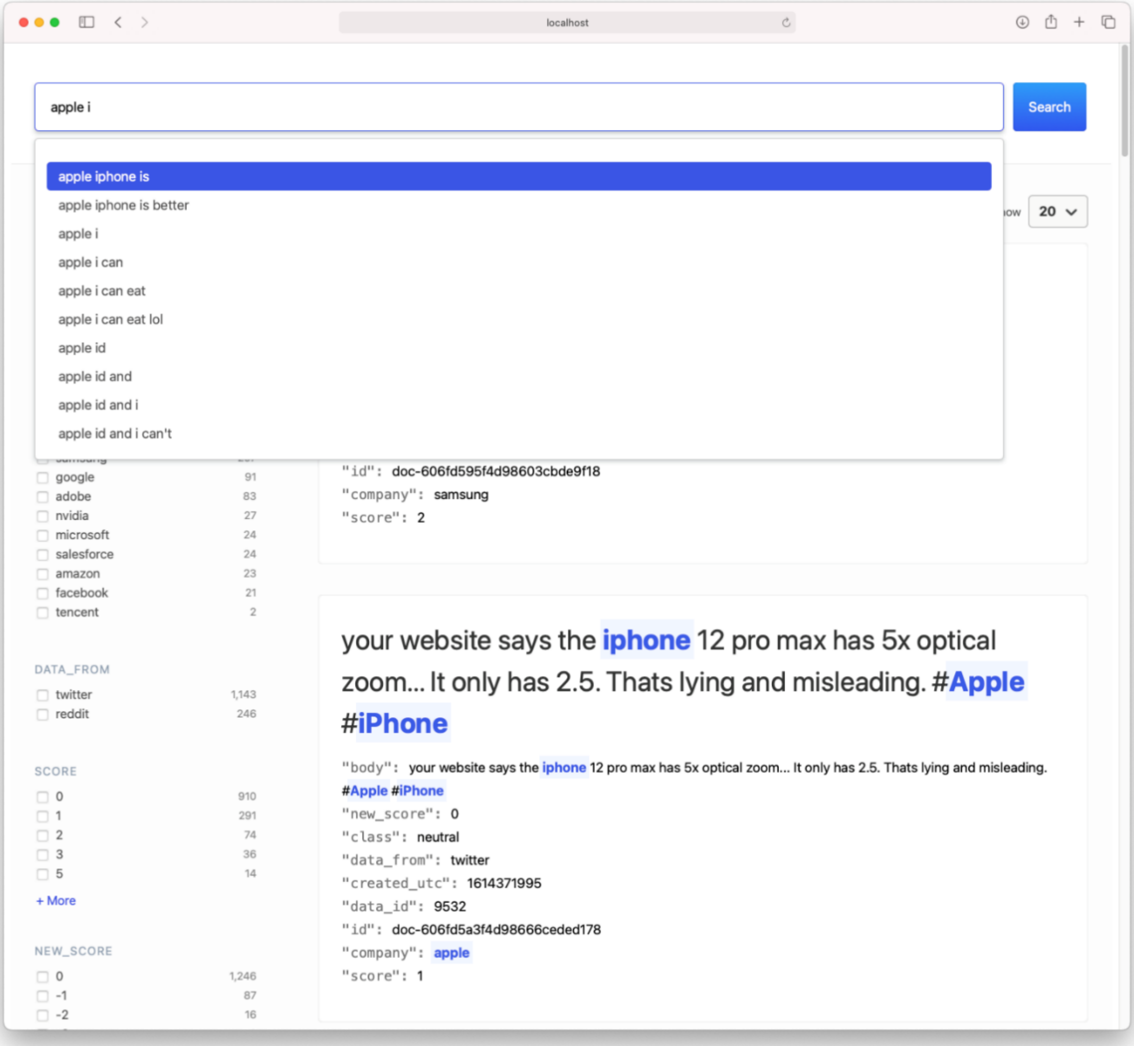
In terms of scalability, the system can be easily scaled under the following steps:

1. Train and save a new model.
2. Load the new model into the existing python code.
3. Add in the result of the prediction with the other models.
4. Calculate new majority.

Upon completion of data modelling, class of all crawled data were predicted and pushed into logstash, using the same JSON format in **3.1 Obtaining Data**. In addition, two new keys were also added: class and new_score. The original score only reflects the magnitude of the weight. For the new_score, it is recalculated based on the predicted label. For example, positive data will have a positive new_score, whereas negative data have negative new_score. This will be useful when businesses chooses to view the new_score in ascending or descending order.

3.5 Interpreting Data





Suggestions are made possible by using Levenshtein distance.

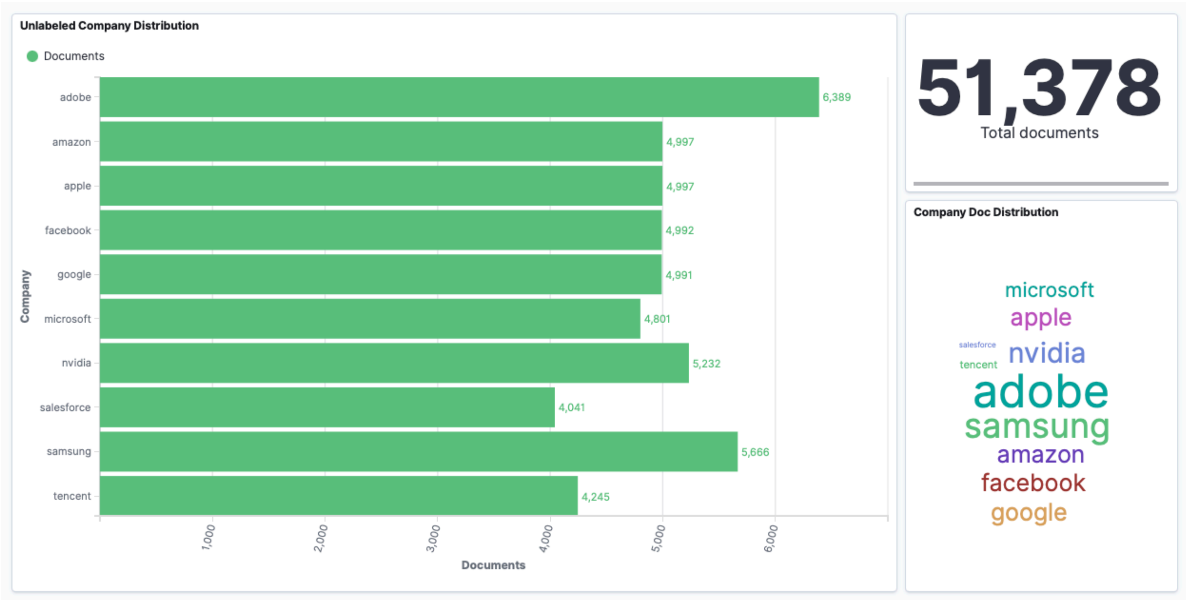
A web search user interface was also made using React.js. User can query using keywords and terms. Example queries and time taken for query to successfully response to the user (see **4.0 Answering Questions in Assignment.pdf** for result returned for the queries)

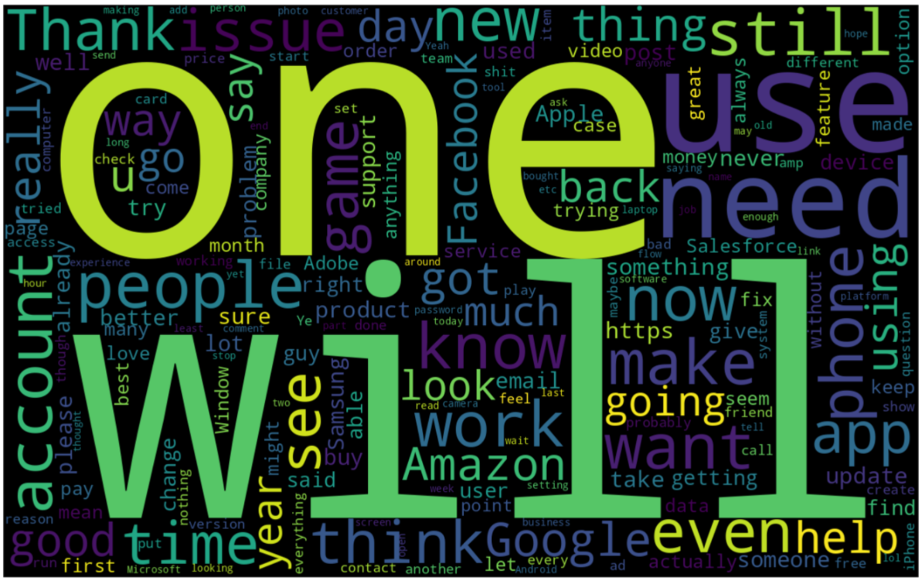
1. google gmail : 142ms
2. facebook marketplace : 130ms
3. apple iPhone : 136ms
4. students email negative : 124ms
5. tencent games : 139ms

4.0 Answering Questions in Assignment.pdf

We answer questions here that we were not able to fit into flow of our report.

Q1.3: The numbers of records, words, and types (i.e., unique words) in the corpus





Q2.3: Write five queries, get their results, and measure the speed of the querying

'google gmail'	Please help me google my gmail account hacked pleas reply	Negative -3	142ms
	Hallo Google teem my Gmail account no recover my password loss	Neutral	
	Dear Google in Gmail, please add an option to delete mails in 'social' category, with one single	Neutral	
	Google services, like browser, Drive, Gmail have become inefficient in MYANMAR due to the Coup that is hindering the access of the internet.	Negative 0	
	? If Google is turning off Gmail accounts because of YouTube issues then I will start telling people to avoid Google accounts and go with Office 365.	Negative -2	

‘facebook marketplace’	I sold it on Kijiji/Facebook Marketplace	Negative 0	130ms
	Hello, kindly help me restore my facebook marketplace . I think I have not committed any mistake Regards Kipkirui kazi	Neutral	
	Ads selling off plots of the Amazon Forest on Facebook marketplace? Why am I not surprised? This	Neutral	
	extremists, disinformation, foreign influence and now this ... Amazon rainforest plots sold via Facebook Marketplace ads	Negative 0	
	save your money people!! don't buy anything off Facebook marketplace if there's ever a discrepancy	Negative 0	

'apple iphone'	Apple iPhone is better	Neutral	136ms
	your website says the iphone 12 pro max has 5x optical zoom... It only has 2.5. Thats lying and misleading. #Apple #iPhone	Neutral	
	why is able to access my location when I have their app set to not sharing my location with them ? #Apple #iPhone #privacy	Neutral	
	Very worst apple iphone since iOS 14 update. Made iPhone worst performance n battery not value of	Negative 0	
	product for you. #Amazon #bestsellers #Chargers #ChargeOn #DigitalMarketing #Apple #iPhone #USA #laptops	Negative 0	

‘student email negative’	very angry parent of 1st year Uni student. He was assaulted + iPhone stolen outside uni halls last	Negative 0	124ms
	"Please let me know if you do not receive this email." as the last line of an email. I call it Schrodinger's email.	Negative -1	
	my sister has had her account hacked. They have removed her email, she has an email from you but	Negative -15	
	Please what's the meaning of this email, do you guys actually sent this mail?	Negative -1	
	How do I reach customer service for email password access?	Negative 0	

	another banger by Riot Games	Negative -17	
	"Please let me know if you do not receive this email." as the last line of an email. I call it Schrodinger's email.	Neutral	
	none. fuck u riot your games suck	Negative 0	
‘tencent games’	You only know how to do this because your games are a real shit and more after fixing games you put	Negative 0	124ms
	For the love of god, please make smurfing in League of legends a bannable offense. It's ruining games.	Neutral 0	

Q4.1 A simple UI for visualizing classified data would be a bonus (but not compulsory)

Using Kibana we were able to quickly build a simple UI that help us visualise the classified data that we obtained from the labelling webapp.

5. Conclusion

In conclusion, we explored and demonstrated the ability to perform the information retrieval process – we obtained the data from online sources, scrubbed it of various impurities, labelled it, created a classifier to automatically label it, and finally visualized the data to obtain a more intuitive understanding of our processed data.