Automated Tagging of Blogs Using LDA and Sentiment Analaysis

Anurag Mishra
Computer Science, NYU Tandon
aa9279@nyu.edu

Advised by: Professor Gustavo Sandoval

ABSTRACT

Blog tagging is a concept where we add extra information usually in form of keywords or tags to the blogs to summarise it quickly and using just a cluster of tags.

In the generation of Machine Learning and AI and the growing need to handle massive dataset has become the need of the hour. Every day almost 5 million blogs are posted and it has shown an incremental pattern year-on-year.

It is understandable how much businesses can achieve if they leave no data untamed. But with such immense data comes the need of technology to handle the kind of data hand.

KEYWORDS

Automated Tagging,Machine Learning, LDA,Sentiment Analysis, Kafka Queue, Zookeeper, Elastic Search, Word2Vec,Query

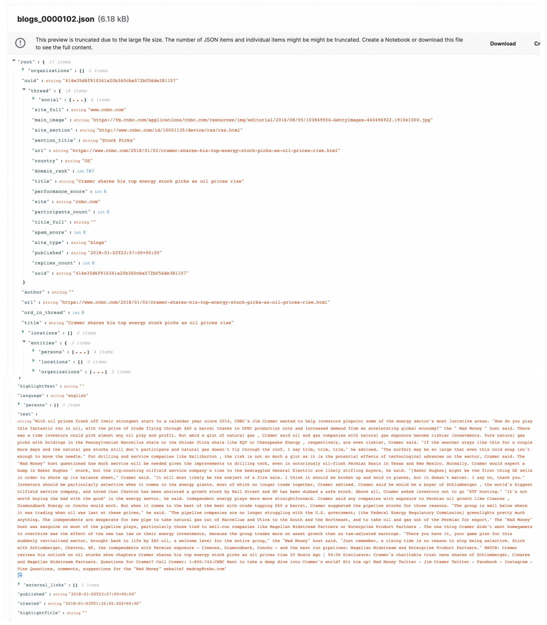
Reference Format:
Anurag Mishra. 2022. Automated Tagging of Blogs Using LDA and Sentiment Analaysis. In *NYU Tandon Capstone Final Project Reports, Spring 2022, New York, USA.* 5 pages.

1 INTRODUCTION

The goal of this project was to generate the tags for the different kinds of blogs or articles. Users can specify the data and the query terms to search for the related blogs. All the related blogs are then be dumped into a Kafka queue which will process the streaming data and generate the associated tags with each blog Users can go to the specific blogs according to their choice. Its also include a pre-processing i.e, word2vec and TF-IDF. After the preprocessing the blogs are tagged using topic modeling- Latent Dirichlet Allocation(LDA) in pySpark. To minimize the storage requirement for the whole system we purge the resulting data after publishing the results to the user. We used pySpark, Apache Kafka and python which will be used to preprocess the massive data.

Capstone Final Project, Spring 2022, New York, USA

Figure 1: Typical Json Tuple from Our Dataset


blogs_0000102.json (6.18 kB)
This preview is truncated due to the large file size. The number of JSON items and individual items might be might be truncated. Create a Notebook or download this file to see the full content.
Download
Cr

```
"root" : { 17 items
  "organizations" : [] 0 items
  "uuid" : string "414e35d6f910341a20b360cba572b056de381107"
  "thread" : { 18 items
    "social" : {...} 6 items
    "site_full" : string "www.cnbc.com"
    "main_image" : string "https://fm.cnbc.com/applications/cnbc.com/resources/img/editorial/2016/08/05/103849554-GettyImages-460496922.1910x1000.jpg"
    "site_section" : string "http://www.cnbc.com/id/10001135/device/rss/rss.html"
    "section_title" : string "Stock Picks"
    "url" : string "https://www.cnbc.com/2018/01/02/cramer-shares-his-top-energy-stock-picks-as-oil-prices-rise.html"
    "country" : string "US"
    "domain_rank" : int 767
    "title" : string "Cramer shares his top energy stock picks as oil prices rise"
    "performance_score" : int 0
    "site" : string "cnbc.com"
    "participants_count" : int 0
    "title_full" : string ""
    "spam_score" : int 0
    "site_type" : string "blogs"
    "published" : string "2018-01-02T23:57:00+00:00"
    "replies_count" : int 0
    "uuid" : string "414e35d6f910341a20b360cba572b056de381107"
  }
  "author" : string ""
  "url" : string "https://www.cnbc.com/2018/01/02/cramer-shares-his-top-energy-stock-picks-as-oil-prices-rise.html"
  "ord_in_thread" : int 0
  "title" : string "Cramer shares his top energy stock picks as oil prices rise"
  "locations" : [] 0 items
  "entities" : { 3 items
    "persons" : [...] 4 items
    "locations" : [] 0 items
    "organizations" : [...] 2 items
  }
  "highlightText" : string ""
  "language" : string "english"
  "persons" : [] 0 items
  "text" :
  string "With oil prices fresh off their strongest start to a calendar year since 2014, CNBC's Jim Cramer wanted to help investors pinpoint some of the energy sector's most lucrative areas. "How do you play this fantastic run in oil, with the price of crude flying through $60 a barrel thanks to OPEC production cuts and increased demand from an accelerating global economy?" the " Mad Money " host said. There was a time investors could pick almost any oil play and profit. But amid a glut of natural gas , Cramer said oil and gas companies with natural gas exposure become riskier investments. Pure natural gas picks with holdings in the Pennsylvanian Marcellus shale or the Ohioan Utica shale like EQT or Chesapeake Energy , respectively, are even riskier, Cramer said. "If the weather stays like this for a couple more days and the natural gas stocks still don't participate and natural gas doesn't fly through the roof, I say trim, trim, trim," he advised. "The surfeit may be so large that even this cold snap isn't enough to move the needle." For drilling and service companies like Halliburton , the risk is not so much a glut as it is the potential effects of technological advances on the sector, Cramer said. The "Mad Money" host questioned how much service will be needed given the improvements to drilling tech, even in notoriously oil-flush Permian Basin in Texas and New Mexico. Normally, Cramer would expect a bump in Baker Hughes ' stock, but the rig-counting oilfield service company's ties to the bedraggled General Electric are likely stifling buyers, he said. "[Baker Hughes] might be the first thing GE sells in order to shore up its balance sheet," Cramer said. "It will most likely be the subject of a fire sale. I think it should be broken up and sold to pieces, but it doesn't matter. I say no, thank you." Investors should be particularly selective when it comes to the energy giants, most of which no longer trade together, Cramer advised. Cramer said he would be a buyer of Schlumberger , the world's biggest oilfield service company, and noted that Chevron has been anointed a growth stock by Wall Street and BP has been dubbed a safe stock. Above all, Cramer asked investors not to go "ETF hunting." "It's not worth buying the bad with the good" in the energy sector, he said. Independent energy plays were more straightforward. Cramer said any companies with exposure to Permian oil growth like Cimarex , Diamondback Energy or Concho would work. But when it comes to the best of the best with crude topping $60 a barrel, Cramer suggested the pipeline stocks for three reasons. "The group is well below where it was trading when oil was last at these prices," he said. "The pipeline companies are no longer struggling with the U.S. government; [the Federal Energy Regulatory Commission] greenlights pretty much anything. The independents are desperate for new pipe to take natural gas out of Marcellus and Utica to the South and the Northeast, and to take oil and gas out of the Permian for export." The "Mad Money" host was sanguine on most of the pipeline plays, particularly those tied to well-run companies like Magellan Midstream Partners or Enterprise Product Partners . The one thing Cramer didn't want homegamers to overthink was the effect of the new tax law on their energy investments, because the group trades more on asset growth than on tax-adjusted earnings. "There you have it, your game plan for this suddenly revitalized sector, brought back to life by $60 oil, a welcome level for the entire group," the "Mad Money" host said. "Just remember, a rising tide is no reason to stop being selective. Stick with Schlumberger, Chevron, BP, the independents with Permian exposure – Cimarex, Diamondback, Concho – and the best run pipelines: Magellan Midstream and Enterprise Product Partners." WATCH: Cramer revises his outlook on oil stocks show chapters Cramer shares his top energy stock picks as oil prices rise 20 Hours Ago | 04:56 Disclosure: Cramer's charitable trust owns shares of Schlumberger, Cimarex and Magellan Midstream Partners. Questions for Cramer? Call Cramer: 1-800-743-CNBC Want to take a deep dive into Cramer's world? Hit him up! Mad Money Twitter - Jim Cramer Twitter - Facebook - Instagram - Vine Questions, comments, suggestions for the "Mad Money" website? madcap@cnbc.com"
  "external_links" : [] 0 items
  "published" : string "2018-01-02T23:57:00+00:00"
  "crawled" : string "2018-01-03T01:32:45.002+00:00"
  "highlightTitle" : string ""
```

2 HYPOTHESIS

"Well-defined tagging of articles can help us retrieve cluster of similar articles pertaining to a keyword which can be done by constructing an article database with tags that capture the essence of the article."

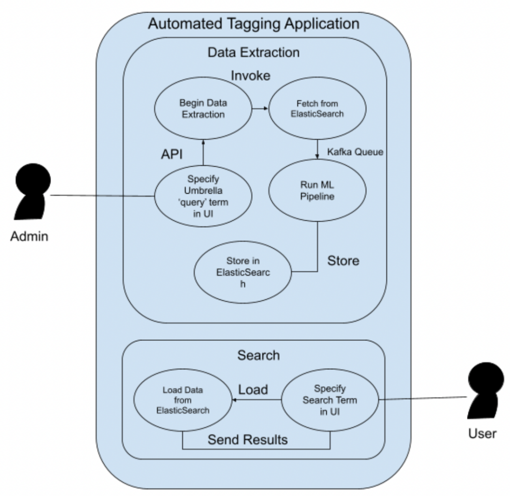
3 SYSTEM DESIGN AND OVERVIEW

3.1 Data Description and Source

The data consists of JSON files which have articles based on the following criteria:

News publishers: Bloomberg.com, CNBC.com, reuters.com, wsj.com, fortune.com Language: Only English Country: United States News Category: Financial news only The source for

Figure 2: Shows the System Overview



the articles (Archive source: httsps://Webhose.io/archive Total Data size is 1GB and the number of json files are 306K in 5 folders each folder for each news channel

3.2 System Flow

The UI allows the user to specify an umbrella query term. This query term is used to gather the relevant documents by querying an elastic search index. All the data collected using elastic search are then dumped into a Kafka queue, which helps in processing the streaming data and managing huge workloads due to the high amount of the relevant articles. The ML pipeline process consumes data from the Kafka queue. Incoming documents are subject to the cleaning of data for further analysis. It then goes through the core of the system for automated tagging. This is achieved by using Latent Dirichlet Allocation (LDA) to generate the tags relevant to each article. We also perform sentiment analysis in each article using NLTK. The generated tags, along with the sentiment, are then dumped into another ElasticSearch index. The UI will fetch the results from the ElasticSearch index and publish them. Now the user can perform searches from the search screen present in the UI and will be presented with the tags relevant to each article. The system also provides a link to every article in case the user wants to read the article.

4 TECHNOLOGIES USED FOR IMPLEMENTATION

(1) Elastic Search Elasticsearch is the distributed search and analytics engine at the heart of the Elastic Stack. Logstash and Beats facilitate collecting, aggregating, and enriching your data and storing it in Elasticsearch. Kibana enables you to interactively explore, visualize, and share insights into your data and manage and monitor the stack. Elasticsearch is where the indexing, search, and analysis magic happens. Elasticsearch provides near real-time search and analytics for all types of data. Whether you have structured or unstructured text, numerical data, or geospatial data, Elasticsearch can efficiently store and index it in a way that supports fast searches.

(2) Kafka Apache Kafka is a community distributed event streaming platform capable of handling trillions of events a day. Initially conceived as a messaging queue, Kafka is based on an abstraction of a distributed commit log. Since being created and open sourced by LinkedIn

Figure 3: System Architecture Diagram

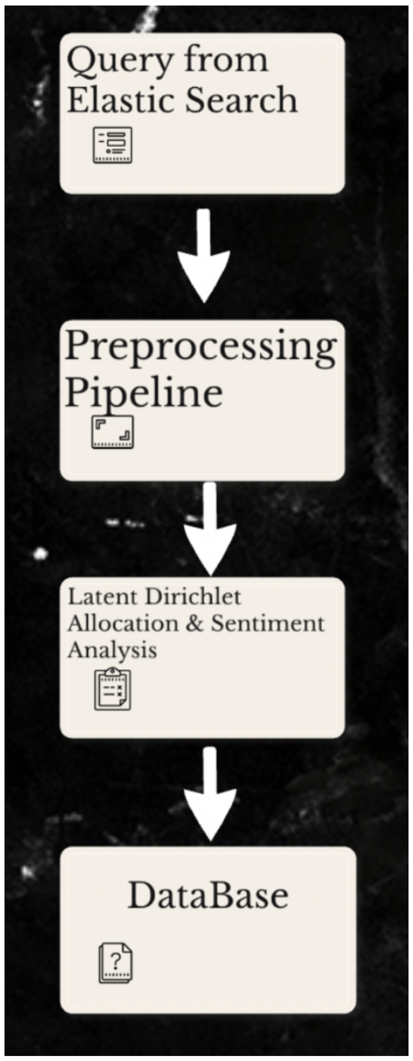
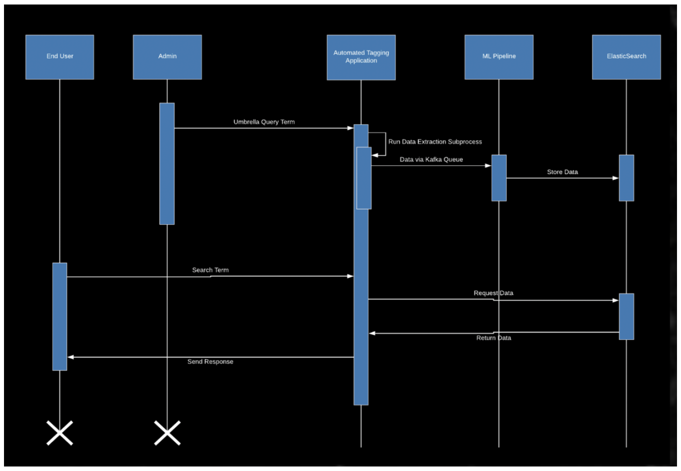

Query from Elastic Search
Preprocessing Pipeline
Latent Dirichlet Allocation & Sentiment Analysis
DataBase

Figure 4: Sequence Diagram



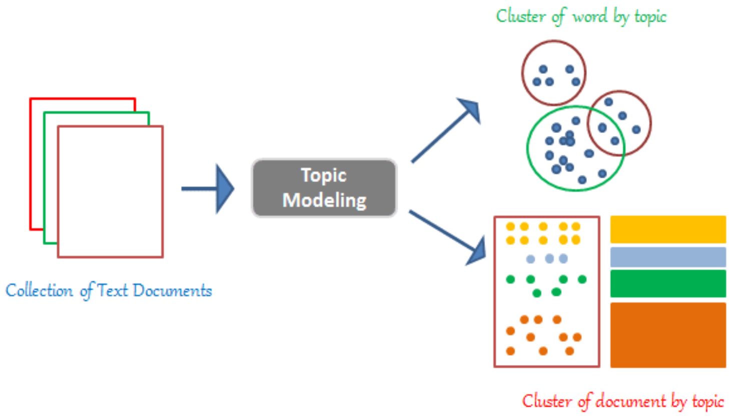
in 2011, Kafka has quickly evolved from messaging queue to a full-fledged event streaming platform.

(3) PySpark PySpark is an interface for Apache Spark in Python. It not only allows you to write Spark applications using Python APIs, but also provides the PySpark shell for interactively analyzing your data in a distributed environment. PySpark supports most of Spark's features such as Spark SQL, DataFrame, Streaming, MLlib (Machine Learning) and Spark Core.

(4) React We wanted our system to be platform-independent and enable the user to access the system from a PC, Mac, or even a smartphone. Our frontend UI is, therefore, built using React. React helps in efficiently updating and rendering the right components of our UI when the data changes. It helps the system deliver beautiful visualizations of the data being processed, enabling the user to make more insightful decisions and ease his process of document discovery and understanding.

(5) Flask Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks. Flask offers suggestions, but doesn't enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that make adding new functionality easy.

Figure 5: A high Level Overview of Latent Dirichlet Allocation



5 MACHINE LEARNING METHODS APPLIED

(1) Latent Dirichlet Allocation(LDA): A statistical model for discovering the abstract topics also known as topic modeling. Topic modeling is a method for unsupervised classification of documents, similar to clustering on numeric data, which finds some natural groups of items (topics) even when we're not sure what we're looking for.

(a) Assumptions:

(i) Each document is just a collection of words or a "bag of words". Thus, the order of the words and the grammatical role of the words (subject, object, verbs, ...) are not considered in the model.

(ii) Words like am/is/are/of/a/the/but/... don't carry any information about the "topics" and therefore can be eliminated from the documents as a preprocessing step. In fact, we can eliminate words that occur in at least For example, if our corpus contains only medical documents, words like human, body, health, etc might be present in most of the documents and hence can be removed as they don't add any specific information which would make the document stand out.

(iii) We know beforehand how many topics we want. 'k' is pre-decided. All topic assignments except for the current word in question are correct, and then updating the assignment of the current word using our model of how documents are generated

5.1 How Does LDA work?

There are 2 parts in LDA:

(a) The words that belong to a document, that we already know.

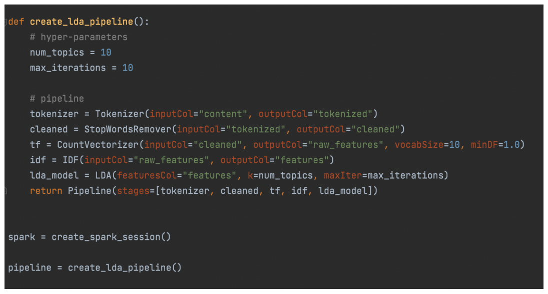


Figure 6: Code snippet of LDA Pipeline Handler

```
def create_lda_pipeline():
    # hyper-parameters
    num_topics = 10
    max_iterations = 10

    # pipeline
    tokenizer = Tokenizer(inputCol="content", outputCol="tokenized")
    cleaned = StopWordsRemover(inputCol="tokenized", outputCol="cleaned")
    tf = CountVectorizer(inputCol="cleaned", outputCol="raw_features", vocabSize=10, minDF=1.0)
    idf = IDF(inputCol="raw_features", outputCol="features")
    lda_model = LDA(featuresCol="features", k=num_topics, maxIter=max_iterations)
    return Pipeline(stages=[tokenizer, cleaned, tf, idf, lda_model])


spark = create_spark_session()

pipeline = create_lda_pipeline()
```

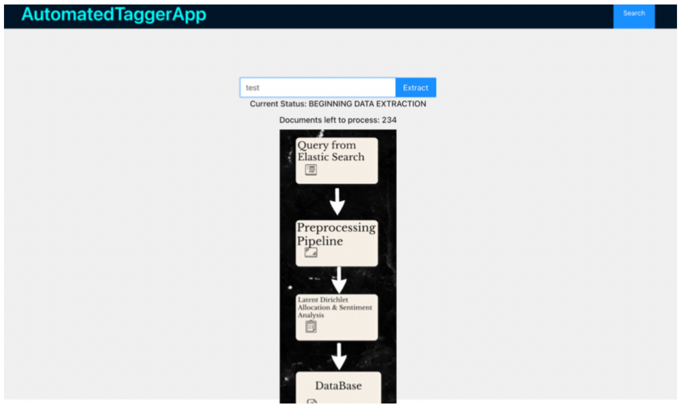


Figure 7: Data Extraction Screen

AutomatedTaggerApp
Search
test
Extract
Query from Elastic Search
Preprocessing Pipeline
Latent Dirichlet Allocation & Sentiment Analysis
DataBase

Current Status: BEGINNING DATA EXTRACTION

Documents left to process: 234

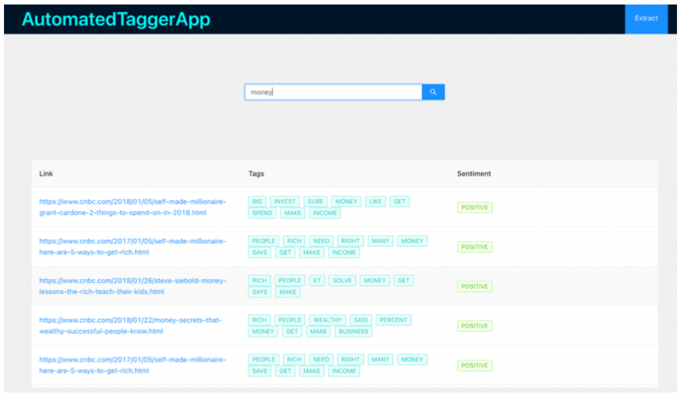


Figure 8: Search Screen

Link	Tags	Sentiment
https://www.cnbc.com/2018/01/05/self-made-millionaire-grant-cardone-2-things-to-spend-on-in-2018.html	BIG INVEST SURE MONEY LIKE GET SPEND MAKE INCOME	POSITIVE
https://www.cnbc.com/2017/01/05/self-made-millionaire-here-are-5-ways-to-get-rich.html	PEOPLE RICH NEED RIGHT MANY MONEY SAVE GET MAKE INCOME	POSITIVE
https://www.cnbc.com/2018/01/26/steve-siebold-money-lessons-the-rich-teach-their-kids.html	RICH PEOPLE ET SOLVE MONEY GET SAYS MAKE	POSITIVE
https://www.cnbc.com/2018/01/22/money-secrets-that-wealthy-successful-people-know.html	RICH PEOPLE WEALTHY SAID PERCENT MONEY GET MAKE BUSINESS	POSITIVE
https://www.cnbc.com/2017/01/05/self-made-millionaire-here-are-5-ways-to-get-rich.html	PEOPLE RICH NEED RIGHT MANY MONEY SAVE GET MAKE INCOME	POSITIVE

(b) The words that belong to a topic or the probability of words belonging into a topic, that we need to calculate.

5.1.1 ***The Algorithm*** . Go through each document and randomly assign each word in the document to one of k topics (k is chosen beforehand). For each document d, go through each word w and compute :

(a) **p(topic t | document d): the proportion of words in document d that are assigned to topic t** tries to capture how many words belong to the topic t for a given document d. Excluding the current word. If a lot of words from d belongs to t, it is more probable that word w belongs to t.
words in d with t +alpha/ words in d with any topic+ k*alpha

(b) **(word w| topic t)**: the proportion of assignments to topic t over all documents that come from this word w. Tries to capture how many documents are in topic t because of word w. LDA represents documents as a mixture of topics. Similarly, a topic is a mixture of words. If a word has high probability of being in a topic, all the documents having w will be more strongly associated with t as well. Similarly, if w is not very probable to be in t, the documents which contain the w will be having very low probability of being in t, because rest of the words in d will belong to some other topic and hence d will have a higher probability for those topic. So even if w gets added to t, it won't be bringing many such documents to t.

(c) Update the probability for the word w belonging to topic t, as

```
p(word w with topic t) = p(topic t | document d) * p(word w | topic t)
```

5.2 Sentiment Analysis

Text sentiment classification is a fundamental sub-area in natural language processing. The sentiment classification algorithm is highly domain-dependent. For example, the phrase "traffic jam" expresses negative sentiment in the sentence "I was stuck in a traffic jam on the elevated for 2 h." But in the domain of transportation, the phrase "traffic jam" in the sentence "Bread and water are essential terms in traffic jams" is without any sentiment. The most common method is to use the domain-specific data samples to classify the text in this domain. However, text sentiment analysis based on machine learning relies on sufficient labeled training data.

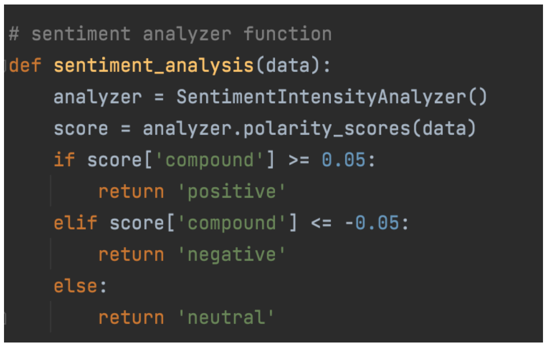
5.2.1 Algorithm for sentiment Analysis. Deciding to not make things complicated I just created a sentiment function which took the probabilty of each word and classifies it based on the if the probabilty was > 0.5 (Positive) or less than 0.5 (negative) and other wise (neutral)

A simple classfier function can be repersented as :

$$C\left(x\right)=\begin{cases}positive; x > 0\cdot 5\\ negative; x < 0\cdot 5.\\ neutral; otherwise\end{cases}$$

where C(x) is the sentiment classifier function

Figure 9: Code for C(x)



```
# sentiment analyzer function
def sentiment_analysis(data):
    analyzer = SentimentIntensityAnalyzer()
    score = analyzer.polarity_scores(data)
    if score['compound'] >= 0.05:
        return 'positive'
    elif score['compound'] <= -0.05:
        return 'negative'
    else:
        return 'neutral'
```

6 DEPLOYMENT

Install all dependencies (pip install -r requirements.txt)

(1) Python >= 3.6
(2) Spark
(3) Kafka
(4) ElasticSearch
(5) NPM or React for UI deployment

subsectionSetup

(1) Start the zookeeper, kafka and elastic search services using the following commands:
  (a) zookeeper-server-start
  (b) /usr/local/etc/kafka/zookeeper.properties
  (c) kafka-server-start /usr/local/etc/kafka/server.properties
  (d) elasticsearch
(2) Extract the project zip
(3) cd back into the project directory and run the following python scripts as shown below
(4) python3 backend.controller.ApiController.py
(5) python3 backend.ldapipeline.backendEngine.py
(6) To load the dataset into elasticsearch, run ‘python -m backend.dataHandler.initES‘
(7) Finally, cd into the UI directory and Start the react app by running the command “ npm start”. Then access the UI on the browser at http://localhost:3000/

7 CONCLUSION AND APPLICATIONS

Undocumented and poorly documented data causes problems for most data teams. Data users would love to have all of their data documented, but the effort to do this is far too large. Building an automated data documentation system to instantly document summarise the data is the need of the hour.

Documenting data is extremely important for any successful data initiative, but most people DESPISE documenting it. Documentation takes time and effort and is often created on the fly resulting in it not being a priority.This tool can this approach by building an automated metadata tagging system to automatically classify data tags based on requirements.

There are many applications already in the run like instagram, twitter etc uses hashtags to store information about a post or tweet. This tools help to do the same by first extracting the keywords from the data and then user searching to find relevant documents.

Not only this, but it has gigantic application in optimising the Search engine where people are wasting time on catalogs and manually finding the keywords to link.

I also think that the app created during this project is not self sufficient for the application stated above but its a proof of concept from where amazing applications can be built.

8 REFERENCES

(1) What is Elasticsearch? | Elasticsearch Guide [7.14]
(2) https://medium.com/technofunnel/apache-kafka-in-5-minutes-c92c43ba3f39
(3) https://www.kaggle.com/jeet2016/us-financial-news-articles/version/1
(4) PySpark Documentation — PySpark 3.1.2 documentation
(5) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8509032/
(6) https://towardsdatascience.com/latent-dirichlet-allocation-lda-9d1cd064ffa2
(7) https://towardsdatascience.com/using-word2vec-to-analyze-news-headlines-and-predict-article-success-cdeda5f14751
(8) https://thinkinfi.com/latent-dirichlet-allocation-for-beginners-a-high-level-overview/