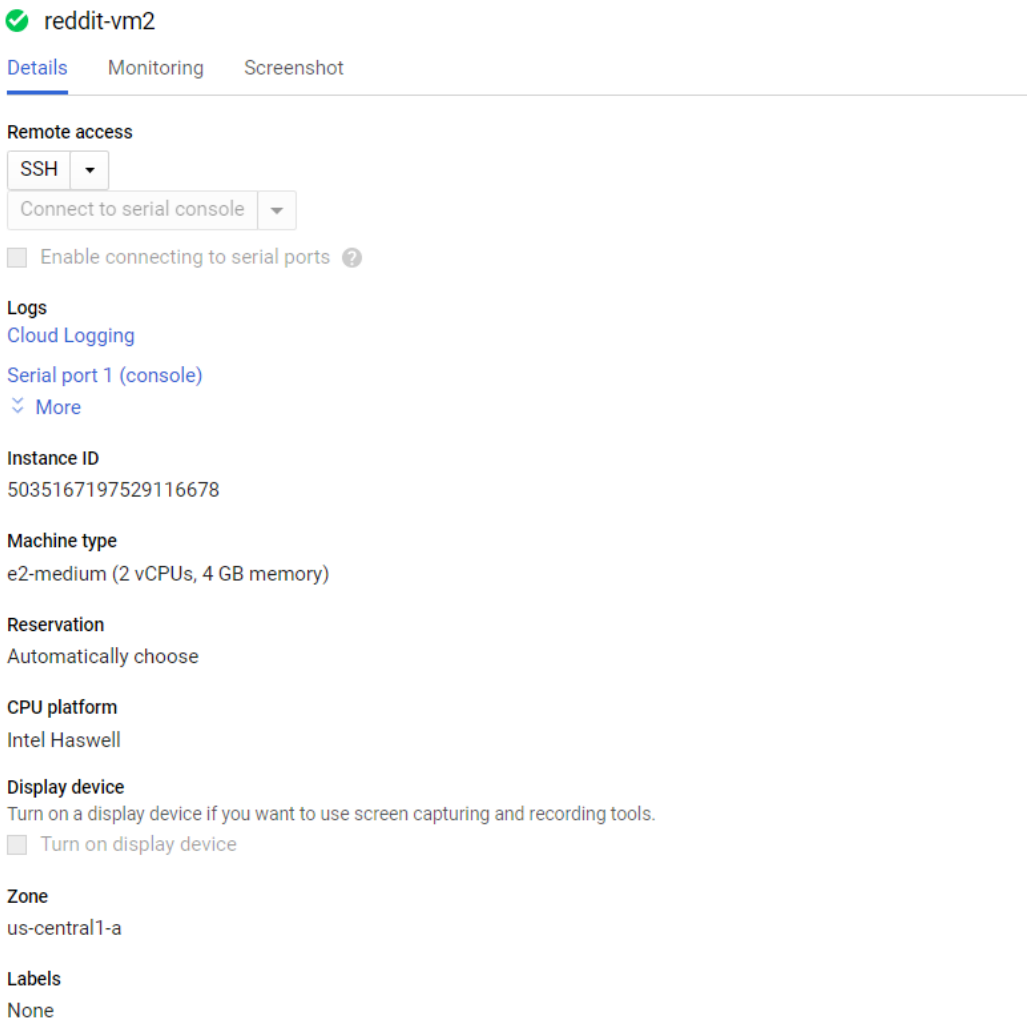
GCP Setup Steps:

Step 1. Data Collection and the Instance: A VM instance is created on google cloud platform and a python script is executed to retrieve replies on a reddit post passed as an URL to the script, by accessing a Python Reddit API Wrapper.



Network interfaces

Name	Network	Subnetwork	Primary internal IP	Alias IP ranges	External IP	Network Tier	IP forwarding	Network details
nic0	default	default	10.128.0.4	–	35.223.34.52 (ephemeral)	Premium	Off	View details

Public DNS PTR Record
None

Firewalls
- [x] Allow HTTP traffic
- [x] Allow HTTPS traffic

Network tags
http-server, https-server

Deletion protection
- [ ] Enable deletion protection

When deletion protection is enabled, instance cannot be deleted. Learn more

Confidential VM service
Disabled

Boot disk

Name	Image	Size (GB)	Device name	Type	Encryption	Mode
reddit-vm2	debian-10-buster-v20201112	10	reddit-vm2	Standard persistent disk	Google-managed	Boot, read/write

Additional disks
None

Local disks
None

Preserved state size
0 GB

Shielded VM
To edit Shielded VM features you need to stop the instance first.

Shielded VM
To edit Shielded VM features you need to stop the instance first.
Turn on all settings for the most secure configuration.
- [ ] Turn on Secure Boot
- [x] Turn on vTPM
- [x] Turn on Integrity Monitoring

Availability policies

Preemptibility	Off (recommended)
On host maintenance	Migrate VM instance (recommended)
Automatic restart	On (recommended)

Custom metadata
None

SSH Keys
- [ ] Block project-wide SSH keys

None

Service account
865778962289-compute@developer.gserviceaccount.com

Cloud API access scopes
Allow full access to all Cloud APIs

Equivalent REST

Figure 1: VM instance details in Google Cloud project

Installations on the VM instance:

<u>Commands to install python3 and pip:</u>

sudo apt update

sudo apt install python3 python3-dev python3-venv

sudo apt-get install python3-pip

<u>Commands to install google-cloud lib and JDK:</u>

sudo pip3 install google-cloud

sudo pip3 install google-cloud-pubsub

Step 2. Data Ingestion: Cloud Pub/Sub provides a staging location Reddit data on its journey towards storage as logs. The publisher application creates and publishes messages to a topic. Subscriber applications create a subscription to a topic to receive messages from it. We have created "analysereddit-topic" topic in Pub/Sub. The Reddit comments are published to this topic based on the event of their arrival. A subscription "gcf-sentiment_cloudfunction-us-central1-analysereddit-topic" is created on the mentioned topic when the cloud function receives this data.

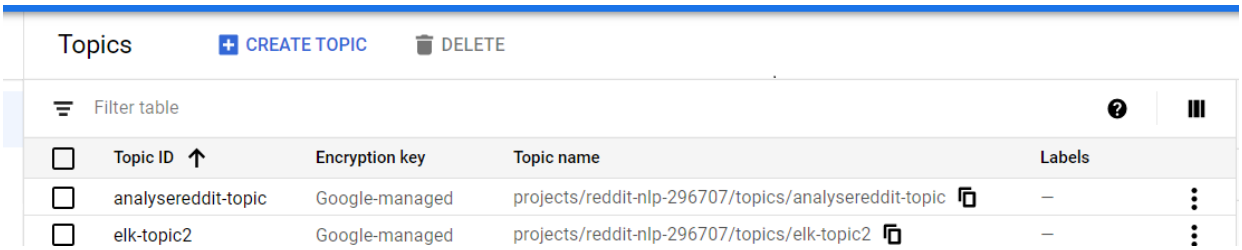


Figure 2: Topics defined in Pub/Sub

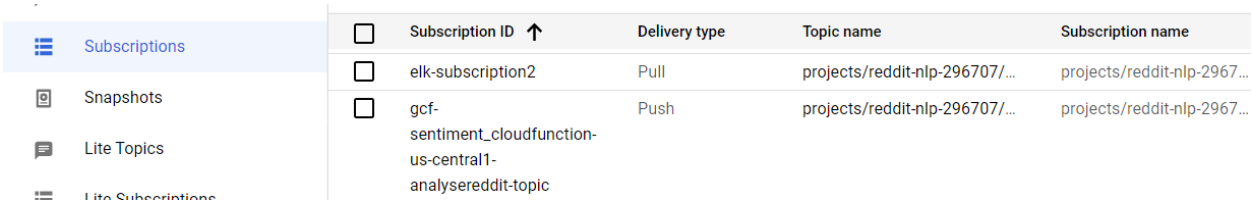


Figure 3: Subscriptions defined in Pub/Sub

Step 3. Sentiment Analysis: Google Cloud Function is an event-driven serverless execution environment where the user's function is attached to the event of Reddit comments published to Pub/Sub. Cloud function invokes Cloud Natural Language API where staged data is converted to a structured JSON format with fields "post", "score", "magnitude".

The results of the sentiment analysis from Google Cloud Function will be stored in the form of Logs.

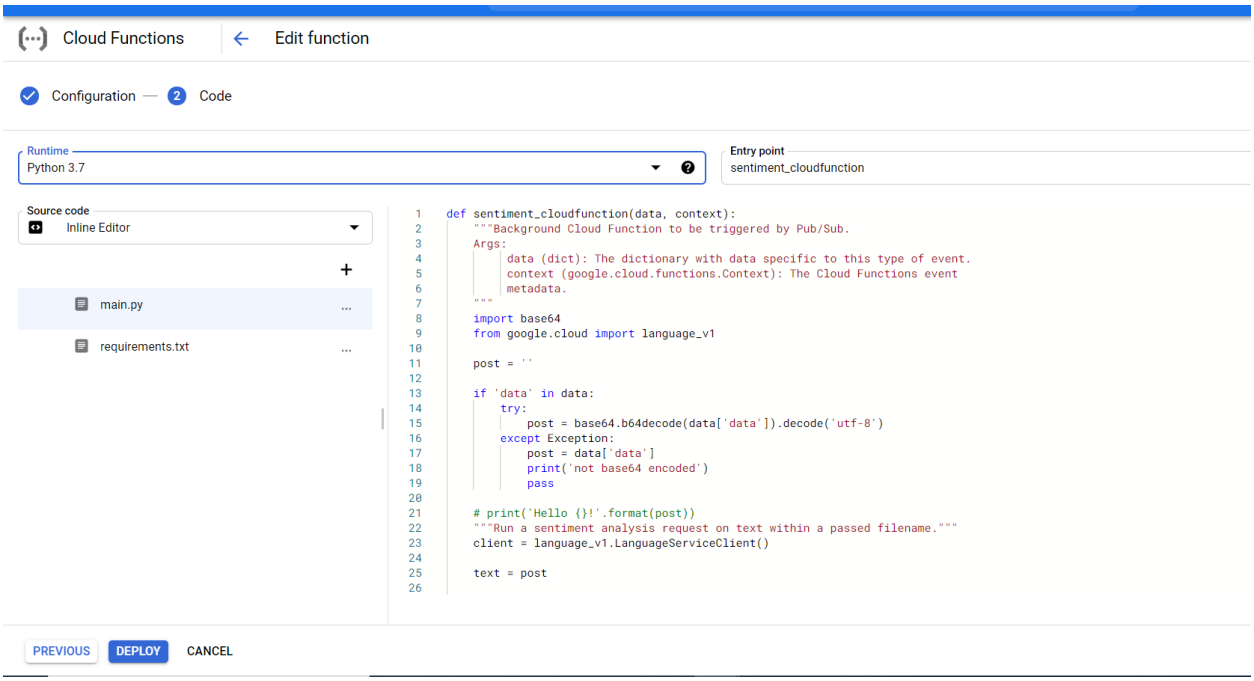


Figure 4: Google Cloud Function deployment

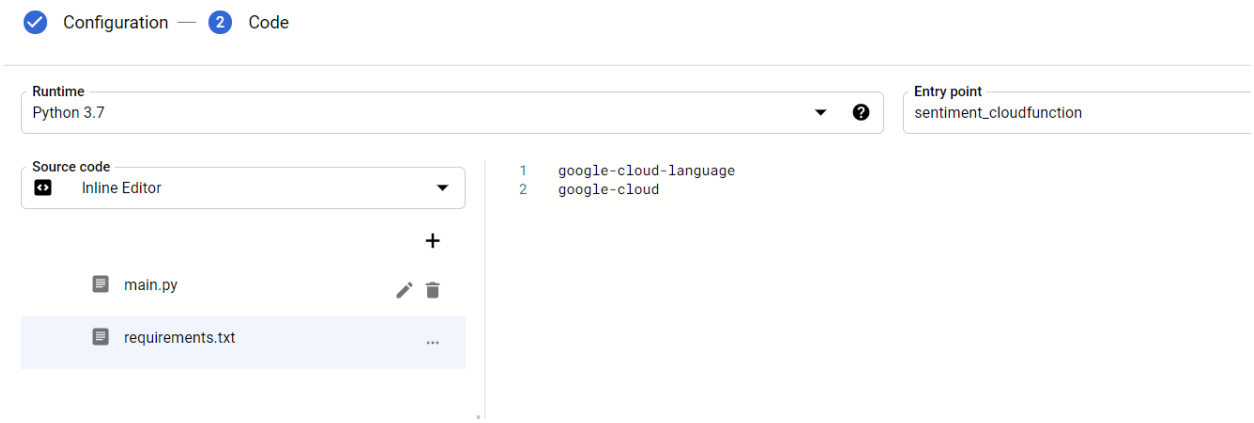


Figure 5: Google Cloud Function requirements.txt file contents

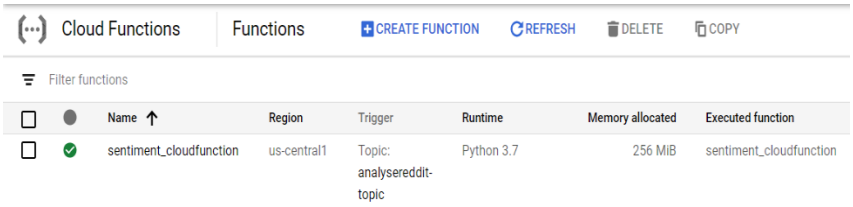


Figure 6: Deployed Google Cloud Function

Command to run the sentiment analysis for Reddit post on VM instance:

(python3 publisher_reddit.py <URL>)

python3 publisher_reddit.py
https://www.reddit.com/r/donaldtrump/comments/k1ltw2/michael_flynn_is_waking_up_a_fully_free_man_after/

Logs created for sentiment analysis of reddit comments:

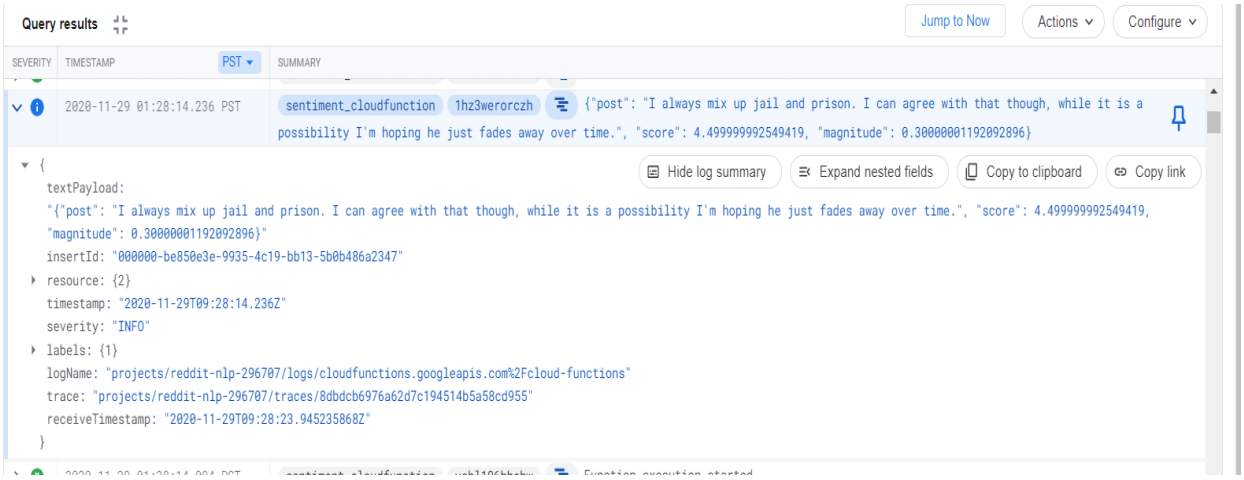


Figure 7: Logs created by Google cloud Function in Operations Logging

Step 4. Storage and Visualization: Pubsubbeats is a type of Beats is an open-source platform for lightweight data shippers which are subscribed to Pub/Sub. Beats can send the data directly to Elasticsearch. In our implementation 'pubsubbeat' elastic beat is subscribed to the topic – elk-topic2. With the help of the 'pull' subscription it will ingest the Google Cloud Logs consisting of sentiment analysis results and send the data to Elasticsearch.

We need to create a 'Sink' in 'Log Router' component of Google Cloud Logging with destination as above topic with inclusion filter of the created cloud Function. This will export the logs to the topic.

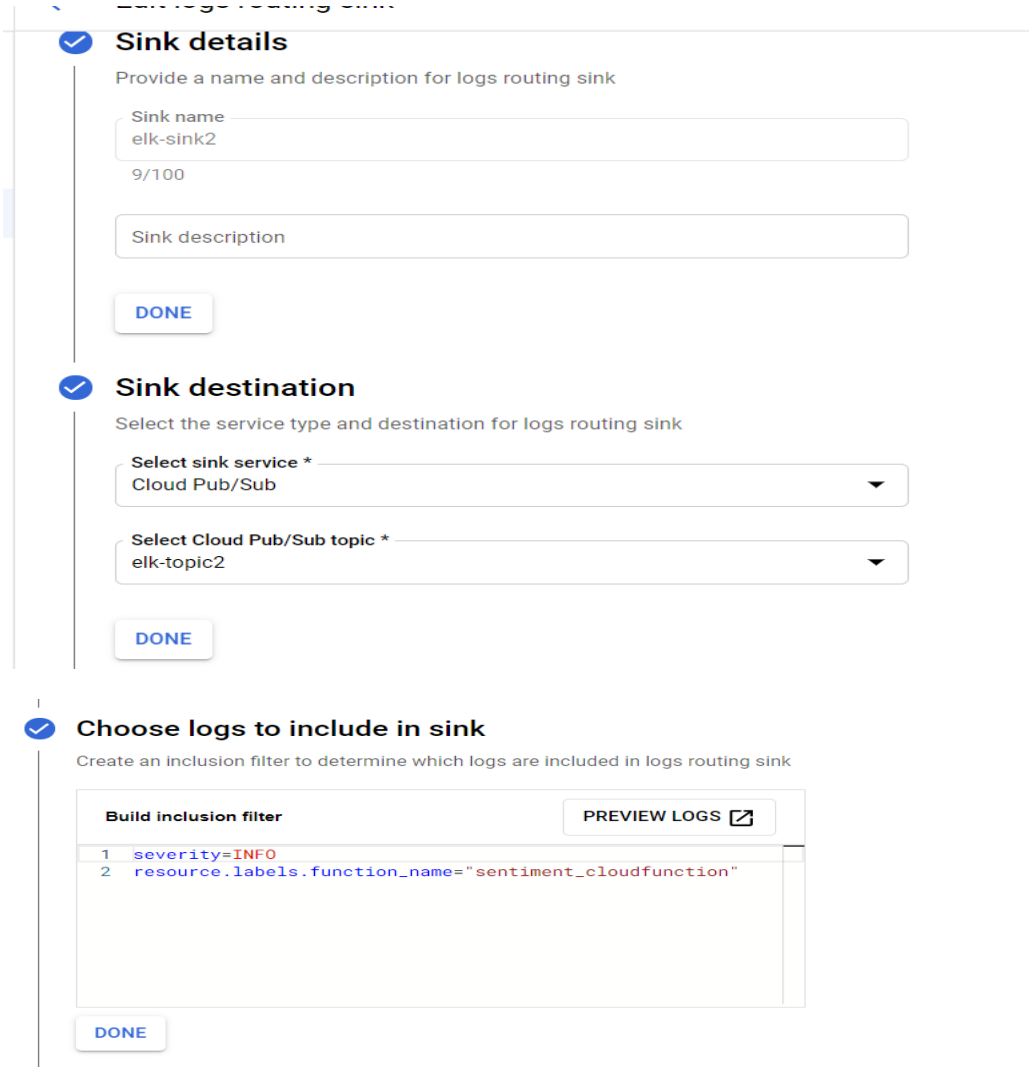


Figure 8: Sink creation in Logs Router.

Installations on VM instance for ELK stack:

<u>Commands to install JDK and nginx:</u>

sudo apt-get update

sudo apt-get install default-jdk

sudo apt-get update

sudo apt-get -y install nginx

sudo systemctl enable nginx

<u>Commands to install Elasticsearch, Kibana and Logstash:</u>

sudo apt-get update

sudo apt-get install wget

sudo wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.2.0-amd64.deb

sudo dpkg -i elasticsearch-7.2.0-amd64.deb

sudo wget https://artifacts.elastic.co/downloads/kibana/kibana-7.2.0-amd64.deb

sudo dpkg -i kibana-7.2.0-amd64.deb

sudo apt-get install -y apt-transport-https

sudo wget https://artifacts.elastic.co/downloads/logstash/logstash-7.2.0.deb

sudo dpkg -i logstash-7.2.0.deb

<u>Command to change elasticsearch.yml Configuration file:</u>

sudo vi /etc/elasticsearch/elasticsearch.yml

Changes: uncomment cluster.name, node.name, path.data, path.logs, network.host, http.port and make sure below are the values:

```
# ---------------------------------- Cluster -----------------------------------
#
# Use a descriptive name for your cluster:
#
cluster.name: my-application
#
# ------------------------------------ Node ------------------------------------
#
# Use a descriptive name for the node:
#
node.name: node-1
#
```

```
path.data: /var/lib/elasticsearch
#
# Path to log files:
#
path.logs: /var/log/elasticsearch
```

```
#
network.host: localhost
#
# Set a custom port for HTTP:
#
http.port: 9200
#
```

Figure 9: elasticsearch.yml file configuration

<u>Command to start elasticsearch:</u>

sudo systemctl start elasticsearch

<u>Command to change kibana.yml Configuration file:</u>

sudo vi /etc/kibana/kibana.yml

<u>Set below values in the yml file:</u>

```
# Kibana is served by a back end server. This setting specifies the port to use.
server.port: 5601

# Specifies the address to which the Kibana server will bind. IP addresses and host
# The default is 'localhost', which usually means remote machines will not be able t
# To allow connections from remote users, set this parameter to a non-loopback addr
server.host: "localhost"
```

Figure 10: kibana.yml file configuration

Start Kibana : sudo systemctl start kibana

sudo apt-get install -y apache2-utils

command to set kibana password:

sudo htpasswd -c /etc/nginx/htpasswd.users kibadmin

sudo vi /etc/nginx/htpasswd.users

Change below file:

sudo vi /etc/nginx/sites-available/default

Add contents as below:

Note: Server name is external IP of the VM instance

```
server {
        listen 80;

         server_name 35.223.34.52;

         auth_basic "Restricted Access";
         auth_basic_user_file /etc/nginx/htpasswd.users;

         location / {
                proxy_pass http://localhost:5601;
                proxy_http_version 1.1;
                proxy_set_header Upgrade $http_upgrade;
                proxy_set_header Connection 'upgrade';
                proxy_set_header Host $host;
                proxy_cache_bypass $http_upgrade;
        }

        location /elastic/ {
                proxy_pass http://localhost:9200;
                proxy_http_version 1.1;
                proxy_set_header Upgrade $http_upgrade;
                proxy_set_header Connection 'upgrade';
                proxy_set_header Host $host;
                proxy_cache_bypass $http_upgrade;
        }
}
```

Figure 11: nginx file configuration

sudo systemctl start nginx

sudo systemctl status nginx

Commands to install pubsubbeat:

sudo wget https://gitlab.com/gitlab-org/pubsubbeat/uploads/0a48a545342f6439a3a95ebef0cdea60/pubsubbeat_1.4.0_linux_amd64.tar.gz

tar -zxvf pubsubbeat-linux-amd64.tar.gz

cd pubsubbeat_1.4.0_linux_amd64

Configure the beat's YAML configuration file:

sudo vi pubsubbeat.yml

(add below details in yml file. Project ID, topic name and subscription name must be mentioned accordingly)

```
################### Pubsubbeat Configuration Example #########################

############################# Pubsubbeat ######################################

pubsubbeat:
  # The service account or refresh token JSON credentials file to use
  # to authenticate API calls.
  #
  # If you don't specify a credentials file, this Beat will use a
  # strategy called Application Default Credentials (ADC) to find your
  # application's credentials. If your Beat runs on Google Cloud Platform,
  # this strategy is recommended.
  #
  # To learn more about ADC:
  # - https://cloud.google.com/docs/authentication/production
  credentials_file:

  # The GCP project ID with your Pub/Sub topic
  project_id: reddit-nlp-296707

  # The Pub/Sub topic name. You must first create this topic.
  topic: elk-topic2

  # The Pub/Sub subscription name.
  subscription.name: elk-subscription2

  # Attempt to create the subscription.
  subscription.create: false # Defaults to true

  # The settings below are used only if the Beat creates the subscription.

  # Whether to retain acknowledged messages. If true, acknowledged messages
  # will not be expunged until they fall out of the RetentionDuration window.
  subscription.retain_acked_messages: false

  # How long to retain messages in backlog, from the time of publish. If
  # retain_acked_messages is true, this duration affects the retention of
  # acknowledged messages, otherwise only unacknowledged messages are retained.
  # Cannot be longer than 7 days or shorter than 10 minutes.
  # Valid time units are "m" and "h". You can also compose them: "2h30m".
  subscription.retention_duration: 168h # Defaults to 7 days

  # How many simultaneous connections the beat will establish to the Pub/Sub endpoint.
  # Pub/Sub streaming pull has a per-subscriber throughput limit,
  # https://cloud.google.com/pubsub/quotas
  # Increasing the pool size will increase the throughput of the beat until
  # a different quota is hit.
  subscription.connection_pool_size: 1
```

```
  # plain text message payload.
  json.enabled: true

  # If this setting is enabled, Pu
  # in case of JSON unmarshaling e
  # but cannot be used.
  json.add_error_key: true
```

```
# Defines if the HTTP endpoint is enabled.
http.enabled: true
```

Figure 12: *pubsubbeat.yml file configuration*

Kibana is used to visualize the data present in Elasticsearch. In order to enable the visualize the log data in Kibana, we have defined and uploaded an ingest pipeline which processes JSON structure string to create JSON objects (for score and magnitude) to Elasticsearch. These JSON objects can be used to create visualizations in Kibana to carry out various analysis. Whenever the Sentiment analysis logs from Google Cloud Platform's logs are received by Beats, they will first get passed through this ingest pipeline and then will be indexed in Elasticsearch.

Define and upload ingest pipeline to Elasticsearch:

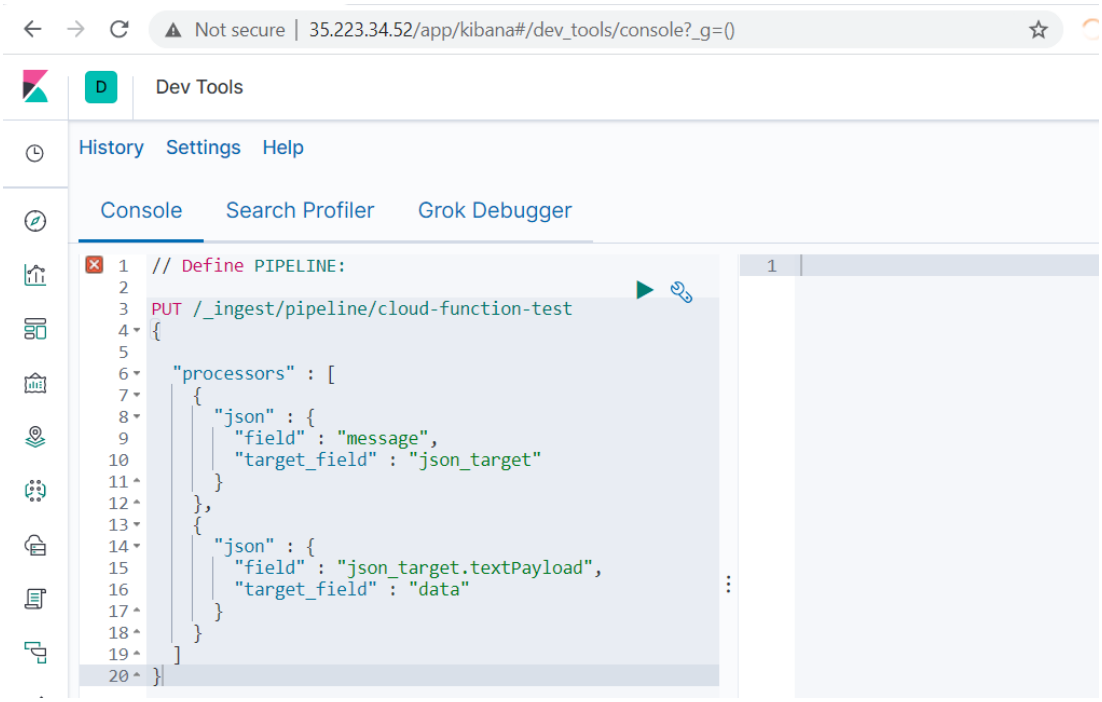


Figure 13: Kibana console: Defining pipeline

This pipeline has two processor nodes to parse out json string into json objects. This will help us to separate out Score and magnitude as independent objects/fields in order to use for visualization and analysis purposes.

Update pubsubbeat.yml to add pipeline for data ingest:

```
#================================ Outputs =====================================

# Configure what output to use when sending the data collected by the beat.

#-------------------------- Elasticsearch output ------------------------------
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
  pipeline: cloud-function-test
  # index: sentiment-logs-test
  #setup.template.name: "sentiment-logs-test"
  #setup.template.pattern: "sentiment-logs-test*"
  # Optional protocol and basic auth credentials.
  #protocol: "https"
  #username: "elastic"
  #password: "changeme"
```

Figure 14: Updation to pubsubbeat.yml after defining ingest pipeline

Once these steps are done, any new data that is received and passed by pubsubbeat, will first go through our pipeline and then will be indexed in elasticsearch.

The main goal of this pipeline is to parse the incoming json string into json objects that can be visualized using kibana.

Commands to start the pubsubbeat:

sudo chown root pubsubbeat.yml

./pubsubbeat -c pubsubbeat.yml -e -d "*"

This step will pull all logs from Google logs and create indexes in Elasticsearch.

Visualizations can be created and viewed in Kibana based on the score for sentiment analysis.

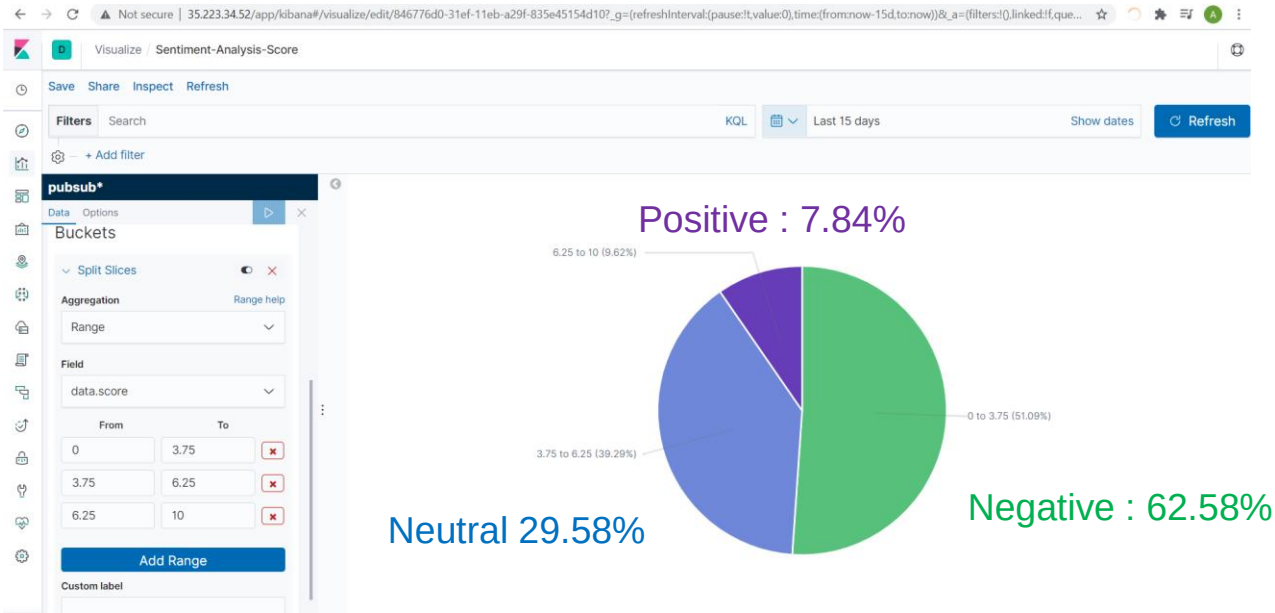


Figure 15: Pie-chart visualization in Kibana for scores for Reddit post