Twitter Sentiment Analysis

AWS Big Data and Machine Learning Services

Ozge Kolcak
Istanbul, Turkey
Bahcesehir University
May, 2019

I. Scenario Definition

Sentiment Analysis is a kind of Natural Language Processing that aims to infer emotional information from a text. It means feelings, emotions, ideas, likes/dislikes and even moods.
In recent years, there has been a growth trend in Sentiment Analysis from brands, companies, researchers and its application to business analytics. Why it is important for marketers is that you can learn the general idea of a customer about your brand in both written and verbal communication. All of these ideas have a great value in data mining. Sentiment analyzes also allow the company to develop a common behavior in order to show the same attitude in the face of similar problems. Considering the volume of data flowing through social media, it is quite difficult to do this with manpower. Therefore, the need for applications that can quickly detect and respond to the positive or negative comments that people write is increasing every day. In this article, I will analyze sentiment data coming from Twitter is a huge psychological database that updates every millisecond.

II. Solution

As all we know that millions of people tweet every day from all over the world with different hashtags or topics. People share their opinion, recommendations, feelings, etc. on Twitter platform. So as to obtain user's feelings expressed in positive or negative comments, I will use AWS Big Data and Machine Learning Services by analyzing a large volume of tweets made with a specific Twitter hashtag.

a. AWS Components Used

Amazon Kinesis Firehose, Lambda, Elasticsearch Service, S3 and Amazon Comprehend as machine learning service have been used to develop this project.
Amazon Kinesis Data Firehose is the effortlessly way to reliably load streaming data into data stores and analytics tools. It can capture, transform, and load streaming data into Amazon S3, Amazon Redshift, Amazon Elasticsearch Service, and Splunk, enabling near real-time analytics with existing business intelligence tools and dashboards.[1]
Amazon Lambda allows running code without provisioning or managing servers. It is only paid for the compute time consumed and if the code does not work, nothing is paid. [2]
Amazon Comprehend is a natural language processing (NLP) service that uses machine learning algorithms to obtain insights from text. [3]
Amazon Elasticsearch Service is a fully managed service that makes it easy for you to deploy, secure, and operate Elasticsearch at scale with zero down time. The service offers open-source Elasticsearch APIs, managed Kibana, and integrations with Logstash and other AWS Services.[4]

b. Other Technologies Used

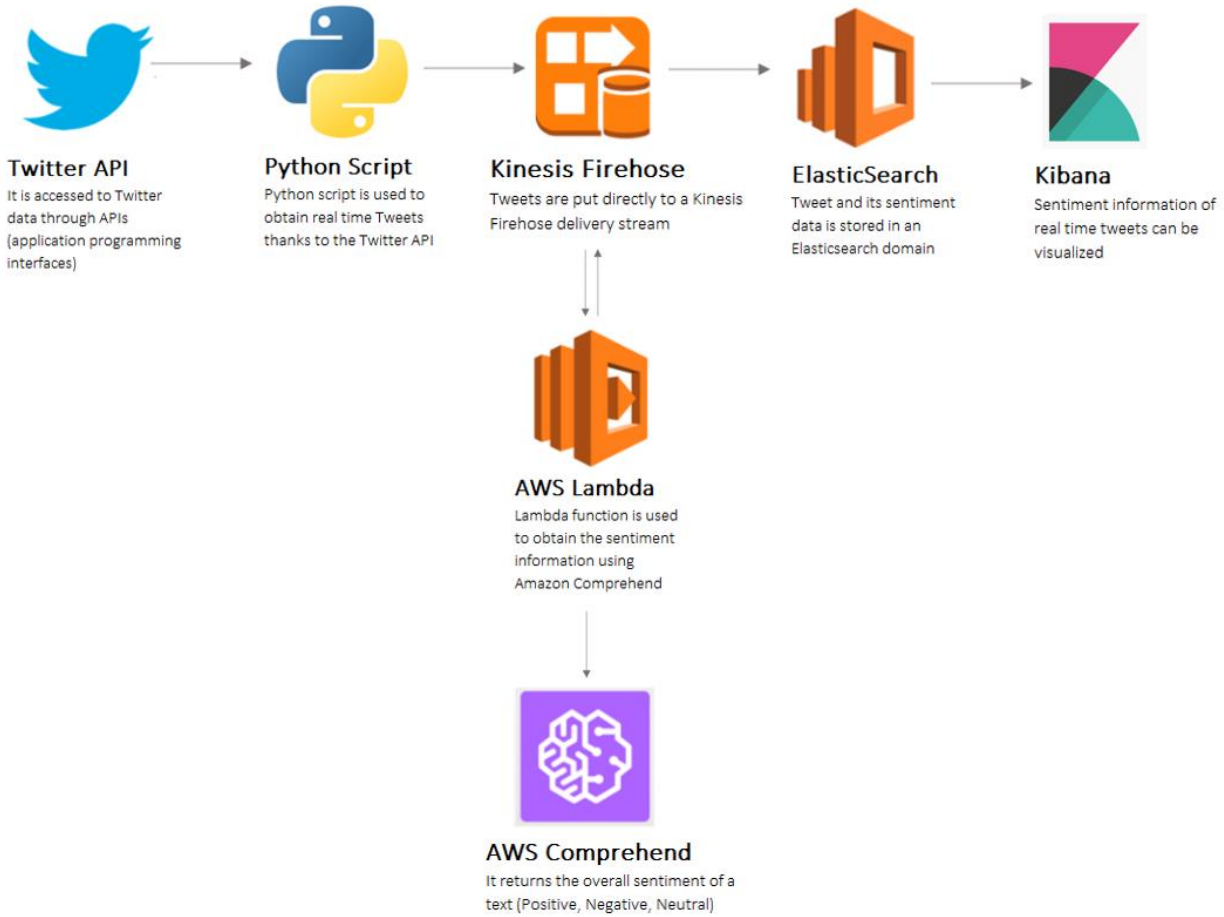
A basic Python script and Twitter development account have been used to obtain real time tweets via Twitter API. It shares information on Twitter as extensively as possible, it also enables companies, developers, and users with programmatic access to Twitter data through APIs (application programming interfaces). [5]

c. Methods Used

In this study, tweets have been pulled from Twitter and identified their sentiment information using AWS Comprehend service. Twitter API has been used to obtain tweets and Python 3.6 version has been used to stream data to Kinesis Firehose delivery stream.

d. Solution Architecture and Design

Firstly, a basic Python script is used to get real time tweets via Twitter API. When the tweets come, they are put directly to a Kinesis Firehose delivery stream that there is a transformation lambda function. After that the sentiment information is obtained using Amazon Comprehend and obtained a clean Twitter comment. Finally, the tweet and its sentiment information are stored in an ElasticSearch domain. We can observe real time tweets data by creating visualizations, dashboards on Kibana.



e. Implementation Details

Twitter Credentials

In order to send Twitter data to Kinesis Firehose, we need to create API keys. First of all, a developer Twitter account is created. After it is approved and it is registered application with Twitter. It is created app at https://apps.twitter.com/ When you create an app, URL is not important that you write dummy. [8] API keys allow to access Twitter account from the Python script using Twitter API (Application Programming Interface). Once this is done we'll have the required API keys to start sending Twitter data to Kinesis Firehose. After that, we will have Consumer API key, Consumer API secret key, Access token, and Access token secret. These keys are also necessary to connect to Twitter account from Python script. We will save these keys by adding AWS key in a config file locally.

Amazon Elasticsearch

A domain named "twitter" is created to store tweet and its sentiment data. And ElasticSearch 6.3 version is chosen. Then "t2.medium.elasticsearch" as instance type and 3 as number of instances are chosen. After public access is set, it is ready to use. [9]

AWS Lambda

Before start to create Lambda function, first we need to create a role:
"Lambda_Comprehend_Role" named that has permissions for ComprehendFullAccess and CloudWatchLogsFullAccess.
And then Lambda_function is created with Python 3.6 Runtime. [11]
A piece of Lambda function is below:

```
comprehend = boto3.client(service_name='comprehend', region_name='eu-west-2)
sentiment_all = comprehend.detect_sentiment(Text=dict_data, LanguageCode='en')
sentiment = sentiment_all['Sentiment']
print(sentiment)
positive = sentiment_all['SentimentScore']['Positive']
negative = sentiment_all['SentimentScore']['Negative']
total = positive - negative
print(total)
```

AWS Comprehend is invoked using 'detect_sentiment' method. [7] After that sentiment type namely Positive, Negative, Neutral is obtained and confidence score based on positive and negative. Most of the tweets have a neutral sentiment, but these tweets also include an amount of positive and negative sentiment; that is why, even if they are a neutral tweet, we can see that little amount of positive or negative inside them with this score.

AWS Kinesis Firehose

Amazon Kinesis Data Firehose is the easiest way to load streaming data into data stores and analytics tools. I've used it for load streaming data into Amazon Elasticsearch Service providing real-time analytics with business intelligence tool Kibana and used for store failed records at S3. Firstly, I've created a new delivery stream "twitter-stream" named and set "Direct PUT" as the source due to using a Python script for it. Then Lambda function previously created is chosen. It provides the record transformation into the stream. Finally, I've set Amazon Elasticsearch Service as destination giving the information of the domain previously created. And in order to

analyze failed records for future, I've set a S3 bucket to store failed records by choosing backup mode as "Failed records only" and "fail-twitter" named. [10]

Python Streaming Code

I've used a Python script to find tweets related to filtering with the word of this specific hashtag. [6] First, I've needed to have pip installed and then pip install tweepy, pip install ConfigParser, and pip install boto3. Also, I've created a config file that includes all the api key in my local. It basically consists of access token, access token secret, consumer key and consumer secret from Twitter api and aws key by creating an IAM user named "ozge". The skeleton of config file is following. The script will connect with the Twitter stream API and filter the information using the written hashtag. And then will put the Tweet's text in the Kinesis Firehose delivery stream.

```
[api_tracker]
access_token=
access_token_secret=
consumer_key=
consumer_secret=
aws_key_id=
aws_key=
```

f. Validation

When everything is set properly and it is fully configured and authorized, it is ready to use all these recently created services with real world tweets coming from Twitter. Python script "twitter-streaming.ipynb" named can kick off the stream and we can see magic tweets flying on the Python script.

I've had an ElasticSearch endpoint with Kibana URL to access data stored. When the Python script is run with a specific hashtag, there will be stored some data in the ElasticSearch. Thus, an index pattern can be created inside Kibana to visualize from this Kibana URL. (Figure 1)

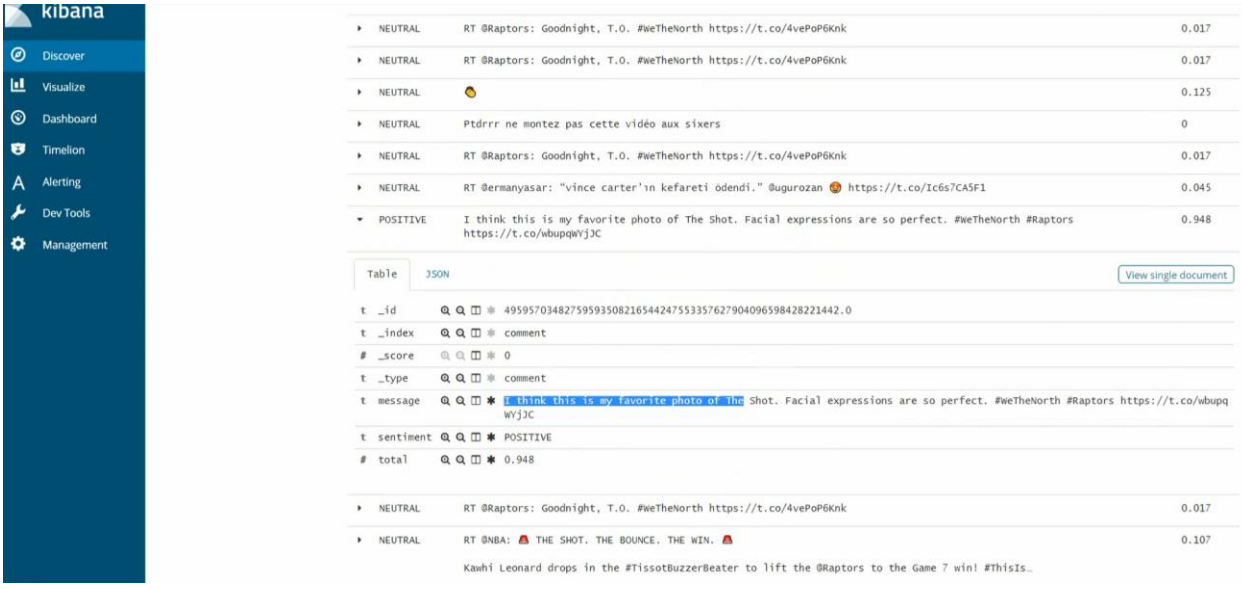


Figure 1

Kibana gives us relevant tweets with #WeTheNorth and its sentiment information as an example. There are some of the tweets showing how the python code literally filtered for this specific hashtag “WeTheNorth”. If we validate the example at Figure 1, we can easily see there are some words such as “favourite”, “perfect”. So AWS Comprehend has been classified this tweet as “Positive”.

I’ve created a pie chart to visualize the number of tweets of each sentiment type: (Figure 2)

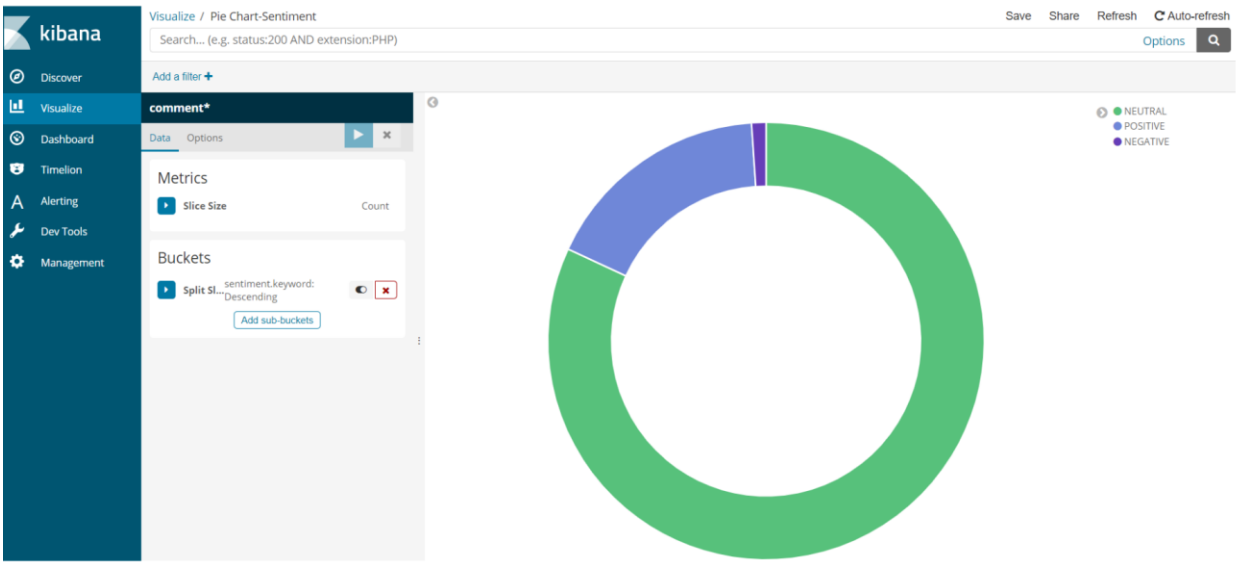


Figure 2

III. Conclusion

Sentiment analysis has been a very popular research area recently. In this study, I've developed a system to evaluate sentiment information of real time tweets from Twitter with AWS Comprehend that is machine learning service. I've first defined the scenario of Twitter Sentiment Analysis. And then I've explained the architecture, configuration, validation respectively using AWS services and Twitter API. When the Python code is run, we have seen the tweets have been filtered with specific hashtag and it has been stored its attributes such as score, message, sentiment, total, id, index, type on Kibana. We have realized that AWS Comprehend Service has been classified tweets properly.

IV. References

[1] https://docs.aws.amazon.com/firehose/latest/dev/what-is-this-service.html

[2] https://aws.amazon.com/lambda/?nc1=h_ls

[3] https://aws.amazon.com/comprehend/?nc1=h_ls

[4] https://aws.amazon.com/elasticsearch-service/?nc1=h_ls

[5] https://help.twitter.com/en/rules-and-policies/twitter-api

[6] https://blogs.sequoiainc.com/twitter-fire-hoses-and-congress-oh-my/

[7] https://docs.aws.amazon.com/comprehend/latest/dg/get-started-api-sentiment.html

[8] https://docs.inboundnow.com/guide/create-twitter-application/

[9] https://docs.aws.amazon.com/elasticsearch-service/latest/developerguide/es-createupdatedomains.html

[10] https://docs.aws.amazon.com/firehose/latest/dev/basic-create.html

[11] https://docs.aws.amazon.com/lambda/latest/dg/resource-model.html