Real Time Tweets Data Analysis Using AWS Kinesis and Apache Spark

Prikshit Batta(20980469), Rahul Sharma(20980992), Sohan Islam(20946046)

University of Waterloo
p2batta@uwaterloo.ca, r364sharma@uwaterloo.ca,s54islam@uwaterloo.ca

Abstract. In this Project, we have built a data pipeline that gets the live Twitter data and that data is ingested into the AWS Kinesis Data stream then AWS Kinesis is connected to Databricks through which the data flows into the data bricks and then it the data is converted into Spark SQL using spark context streaming and spark SQL data frames. After that we will store this data in an S3 bucket and do the sentiment analysis, most trending hashtags, most tagged person, and most hot topics of this data. Thereafter do some transformationa and claning then store this data in an S3 bucket and then we will use the Machine learning Model Naive Bayes Classification to check whether the data fall into the subject of withering science or arts.

Keywords: Twitter, Tweepy, AWS Kinesis, S3 Bucket, Databricks, Apache Spark, Data Streaming

1 Introduction

Social media analysis is a crucial part of machine learning and Data engineering. Twitter is a popular social media platform where people are sharing their statements in terms of their opinion about trending topics AKA tweets. There are a handsome amount of branding companies that are spending millions on social media analysis for their product opinion and sentiment analysis. Also, there are thousands of people who influence the masses (celebrities, politicians, etc.) are twitting Petabytes of data on a daily basis.Twitter provides an API to stream real-time tweets from a particular region or from specific users. With sentiment analysis, we can detect if a piece of text conveys a positive or a negative message. It is called polarity, and this is a game-changer when it comes to gathering consumer feedback from a massive amount of data in a short stretch of time. Using text classifiers, we can automatically structure all manner of relevant text, from emails, legal documents, social media, surveys, and more in a fast and cost-effective way. This allows we can save time analyzing text data, automate business processes, and make data-driven business decisions.

2 Literature Review

Batch processing is when the processing and analysis happens on a set of data that have already been stored over a period of time. An example is payroll and billing systems that have to be processed weekly or monthly.

Streaming data is data that is generated continuously by thousands of data sources, which typically send in the data records simultaneously, and in small sizes (order of Kilobytes). Streaming data includes a wide variety of data such as log files generated by customers using your mobile or web applications, ecommerce purchases, in-game player activity, information from social networks, financial trading floors, or geospatial services, and telemetry from connected devices or instrumentation in data centers.

2.1 Processing of streaming Data

Streaming data needs to be processed sequentially and incrementally on a record-by-record basis or over sliding time windows, and used for a wide variety of analytics including correlations, aggregations, filtering, and sampling. Information derived from such analysis gives companies visibility into many aspects of their business and customer activity such as –service usage (for metering/billing), server activity, website clicks, and geo-location of devices, people, and physical goods –and enables them to respond promptly to emerging situations. For example, businesses can track changes in public sentiment on their brands and products by continuously analyzing social media streams, and respond in a timely fashion as the necessity arises. In this project we will be using twitter data for streaming as illustrated in figure 1.

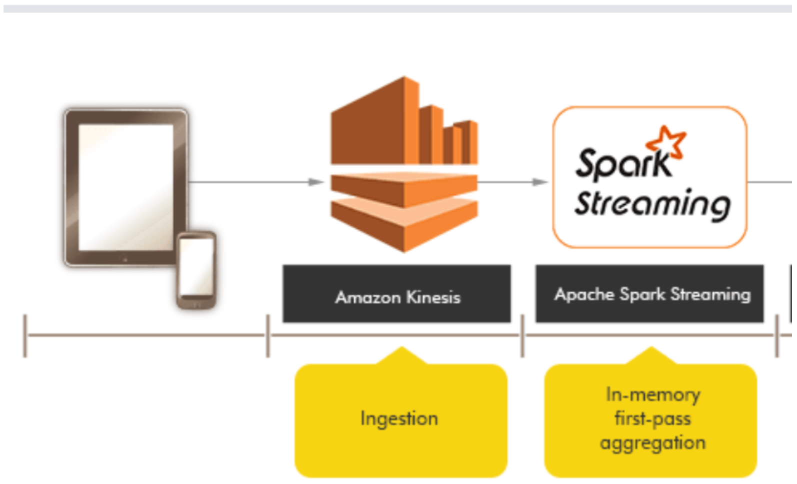


Fig. 1. Data Being Added to AWS Kinesis

2.2 Real-Time Data Streaming Tools

1. *Amazon Kinesis:* Amazon Kinesis is also a notable mention among the top real-time data streaming tools which allows streaming Big Data with AWS. Enterprises can develop streaming applications by leveraging open-source Java libraries and SQL editor with Amazon Kinesis. The best thing about Kinesis is that it takes care of the major responsibilities of running applications and scaling them according to requirements.
   As a result, enterprises can easily reduce the need for managing servers and other complexities related to the development, integration, and management of applications for real-time analytics.
   One of the most crucial traits of Amazon Kinesis that makes it one of the top open-source data streaming tools is flexibility. The flexibility of Kinesis helps enterprises start initially with basic reports and insights on data. Subsequently, with the growth of demand, Kinesis can help in the deployment of machine learning algorithms to support in-depth analysis.

2. *Apache Kafka:*Apache Kafka is also a leading mention among real-time data streaming tools. Enterprises can use Apache Kafka for the management of peak data ingestion loads and also as a big data message bus. The capability of Apache Kafka to manage peak data ingestion loads is a unique and formidable advantage over common storage engines.
   The general application of Kafka is in the back end for the integration of microservices. In addition, it can also support other real-time data streaming portals such as Flink or Spark. Interestingly, the majority of real-time data streaming platforms can integrate effectively with Kafka to provide stream analytics and stream processing.

3. *Azure Stream Analytics:* The design of Azure Stream Analytics focuses on the delivery of mission-critical end-to-end analytics services. Interestingly, Azure Stream Analytics provides faster analytics outcomes within a limited time by leveraging C#, SQL, and JavaScript.
   The in-built machine learning capabilities of Azure Stream Analytics also provide adequate support for intuitive data processing. The machine learning capabilities also help in easier identification of spikes and dips, slow positive and negative trends, and outliers pertaining to streamed data. As a result, users could easily interpret the output visualizations.

   There are number of other tools which can be used for Data streaming puroses like APache NIFi, Stream Sql, Google Cloud Dataflow. We are using Aws Kinesis in our project due to its popularity and good support provided by Amazon.

2.3 Tools used for Storing Data

1. *S3 Bucket:* Amazon Simple Storage Service (Amazon S3) is an object storage service offering industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can store and protect any amount of data for virtually any use case, such as data lakes, cloud-native applications, and mobile apps.

With cost-effective storage classes and easy-to-use management features, you can optimize costs, organize data, and configure fine-tuned access controls to meet specific business, organizational, and compliance requirements. We have used s3 in this project to store the data for certain time period and then take the csv file of that data and done the analysis of that.

2. *Google Bigquery:* Google offers a fully-managed enterprise data warehouse for analytics via its BigQuery product. The solution is serverless and enables organizations to analyze any data by creating a logical data warehouse over managed, columnar storage, and data from object storage and spreadsheets. BigQuery captures data in real-time using a streaming ingestion feature, and it's built atop the Google Cloud Platform. The product also provides users the ability to share insights via datasets, queries, spreadsheets, and reports.

2.4 Spark Streaming

Apache Spark Streaming is a scalable fault-tolerant streaming processing system that natively supports both batch and streaming workloads. Spark Streaming is an extension of the core Spark API that allows data engineers and data scientists to process real-time data from various sources including (but not limited to) Kafka, Flume, and Amazon Kinesis. This processed data can be pushed out to file systems, databases, and live dashboards.

Its key abstraction is a Discretized Stream or, in short, a DStream, which represents a stream of data divided into small batches. DStreams are built on RDDs, Spark's core data abstraction. This allows Spark Streaming to seamlessly integrate with any other Spark components like MLlib and Spark SQL. Spark Streaming is different from other systems that either have a processing engine designed only for streaming, or have similar batch and streaming APIs but compile internally to different engines. Spark's single execution engine and unified programming model for batch and streaming lead to some unique benefits over other traditional streaming systems. In this project we will be using spark streaming to do analysis of the data received from twitter.

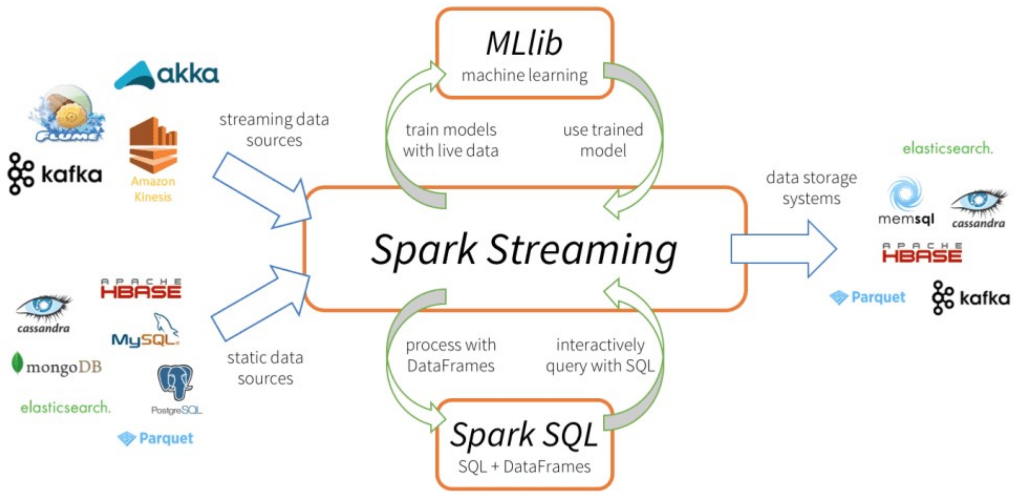


Fig. 2. Apache Spark Streaming

3 Problem Statement

For the analysis of the data, we must have the data in the data set or in the live stream. If we process the data from the dataset then the processing is done in the batches. But the problem here is that how should we analyze the data that is real-time. So one of the solutions to this is real-time processing using Apache spark.

4 Our Contribution

In this project we have built a data pipeline using popular big data platforms, Spark and AWS, to achieve sentiment and time series analysis on Twitter. The pipeline would transform the tweets filtered by hashtags and keywords, into the streaming data format, which can be manipulated in Spark. Once the data is received we can convert the received data into spark SQL schema and then apply the spark RDD data transformation and data preprocessing. After converting the data into SQL data frame we gave the name to the columns as "id, ts, tweet" where id represents the tweet id, ts represents the tweet timestamp, and tweet represents the text in that particular tweet.

In the next step we have converted it into RDD and done data cleaning and tokenized the text. After this we have applied the transformations and action on that data to find the Trending HashTags,Most Tagged Twitter Accounts, Sentiment Analysis, Percentage of Positive, Negative and Neutral Sentiments in the data and Hot words from that data.

We also used the tokenized text on labeled data to train a naive Bayes classifier. For this analysis, we take two instances of label data labeled as Science and Arts. Then after prepossessing the data using PySpark, we applied the model described in Section 5 on the data (code attached in name of "Machine_learning_code"). The implementation of the model and the result is presented in Section 6.2.

4.1 Why Spark Streaming is used?

Like Python, Apache Spark Streaming is growing in popularity. Spark Streaming is better than traditional architectures because its unified engine provides integrity and a holistic approach to data streams.When combined, Python and Spark Streaming work miracles for market leaders. Our main focus here is near real time data streaming as we are getting tweets from twitter so we need any messaging queue to support this. Live input comes into Spark Streaming, and Spark Streaming separates the data into individual batches. These batches are put into the Spark Engine, which creates the final result stream in batches. After we receive the data we can do any type of transformation and data analysis on that. As we have done the sentimental analysis, found the most trending hashtags, found the most tagged Twitter accounts from our data and after we store in the S3 bucket then we have read taken the data from the S3 bucket and applied the Machine learning model to do predictions on that data.

5 Implementation

The essential thing to implement for the project was to find out how to take twitter data and how to send the live stream in AWS Kinesis. We have used tweepy library of python to get the tweets from twitter and then using the boto3 module we have connected with AWS Kinesis and made the pipeline to send the data from twitter api to AWS Kinesis.

5.1 High Level Diagram

In figure no. 3 the architecture of data pipeline is explained below. There are different components in the architecture and each component is described below.

• *Tweepy:* Tweepy is an open source Python package that gives you a very convenient way to access the Twitter API with Python. Tweepy includes a set of classes and methods that represent Twitter's models and API endpoints, and it transparently handles various implementation details, such as: Data encoding and decoding. The role of tweepy library here is to get the tweets from twitter using twitter api.

• *EC2 Instance:* An Amazon EC2 instance is a virtual server in Amazon's Elastic Compute Cloud (EC2) for running applications on the Amazon Web Services (AWS) infrastructure. It provides scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 eliminates your need to invest in hardware upfront, so you can develop and deploy applications faster. In this project we have used the EC2 instance to run the code that is fetching tweets from the twitter api on EC2 instance.

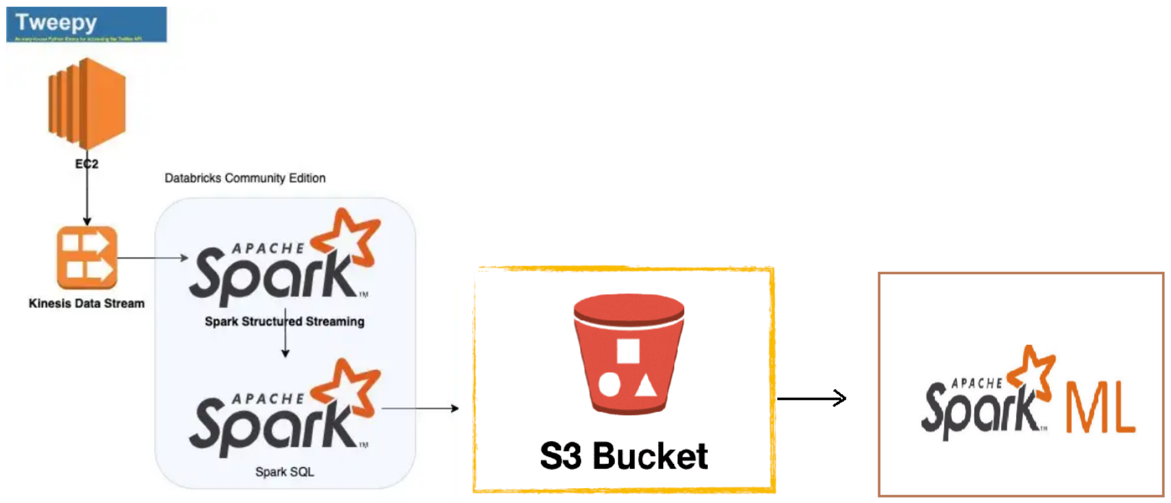


Fig. 3. Architecture of the Data Pipeline

• *Kinesis Data Stream:* Amazon Kinesis Data Streams is a fully managed streaming data service. You can continuously add various types of data such as clickstreams, application logs, and social media to a Kinesis stream from hundreds of thousands of sources. Within seconds, the data will be available for your Kinesis Applications to read and process from the stream. After we get the tweets from twitter we send it to AWS Kinesis data stream.

• *Apache Spark Structured Streaming:* Apache Spark Structured Streaming is a near-real-time processing engine that offers end-to-end fault tolerance with exactly-once processing guarantees using familiar Spark APIs. Structured Streaming lets you express computation on streaming data in the same way you express batch computation on static data. The Structured Streaming engine performs the computation incrementally and continuously updates the result as streaming data arrives.

• *Apache Spark SQL:* Spark SQL brings native support for SQL to Spark and streamlines the process of querying data stored both in RDDs (Spark's distributed datasets) and in external sources. Spark SQL conveniently blurs the lines between RDDs and relational tables.In this project, we have used spark SQL to convert the data to data frame and assign the schema to the data.

• *AWS S3 Bucket:* An Amazon S3 bucket is a public cloud storage resource available in Amazon Web Services' (AWS) Simple Storage Service (S3), an object storage offering. Amazon S3 buckets, which are similar to file folders, store objects, which consist of data and its descriptive metadata. As after the data is processed by data spark SQL we applied transformations and actions on that data and stored the data in an S3 bucket.

• *Databricks Community Edition:* The Databricks Community Edition is the free version of our cloud-based big data platform. Its users can access a micro-cluster as well as a cluster manager and notebook environment. All users can share their notebooks and host them free of charge with Databricks. We have used Databricks community edition to run the spark cluster and get the data from AWS kinesis and perform the spark SQL and other transformations on data.

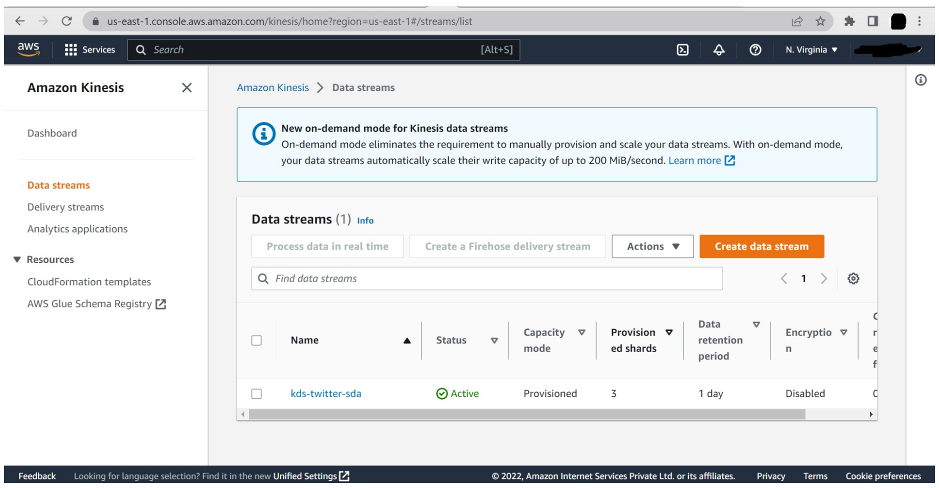
• *Apache Spark ML:* MLlib is Spark's machine learning (ML) library. Its goal is to make practical machine learning scalable and easy. At a high level, it provides tools such as: ML Algorithms: and common learning algorithms such as classification, regression, clustering, and collaborative filtering. As our data gets stored in the s3 bucket as CSV file and now is the time to apply the machine learning model to that data. We applied the Naive Bayes Classifier to the data and checked the accuracy of the data.

5.2 AWS infrastructure Setup

our first step is the infrastructure setup of the services required from AWS. We require the AWS EC2 instance, AWS Kinesis Data stream, and AWS S3 Bucket for this project.

• *Creating EC2 Instance:* we will create an EC2 instance on which we will run our TwitterCalling.py code.

• *Creating Kinesis Data Stream:* We have created Kinesis Data stream named "kds-twitter-sda" in which data will be pushed after tweepy gets from twitter.The Fig 4 shows that data stream is created.



us-east-1.console.aws.amazon.com/kinesis/home?region=us-east-1#/streams/list

aws Services Search [Alt+S] N. Virginia [redacted]

Amazon Kinesis

Dashboard

Data streams
Delivery streams
Analytics applications

Resources
CloudFormation templates
AWS Glue Schema Registry

Amazon Kinesis > Data streams

New on-demand mode for Kinesis data streams
On-demand mode eliminates the requirement to manually provision and scale your data streams. With on-demand mode, your data streams automatically scale their write capacity of up to 200 MiB/second. Learn more

Data streams (1) Info

Process data in real time | Create a Firehose delivery stream | Actions | Create data stream

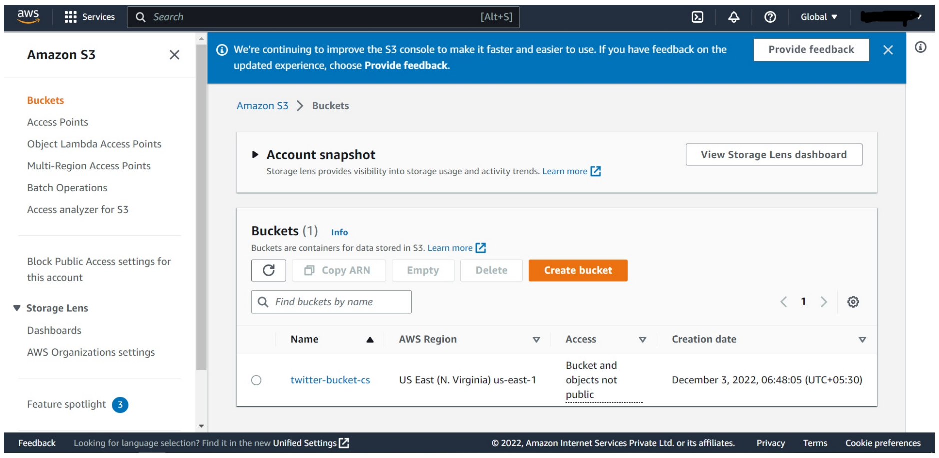
Find data streams

Name	Status	Capacity mode	Provisioned shards	Data retention period	Encryption	C
kds-twitter-sda	Active	Provisioned	3	1 day	Disabled	C

Feedback Looking for language selection? Find it in the new Unified Settings © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

Fig. 4. Data Stream Created

• *Creating S3 Bucket:* We have created an S3 bucket named "twitter-bucket-cs" in which the cleaned data will be stored after preprocessing and then which will be used for machine learning alogorithms.The Fig 5 shows that S3 bucket is created.



aws Services Search [Alt+S] Global [redacted]

Amazon S3

Buckets
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
Access analyzer for S3

Block Public Access settings for this account

Storage Lens
Dashboards
AWS Organizations settings

Feature spotlight 3

We're continuing to improve the S3 console to make it faster and easier to use. If you have feedback on the updated experience, choose **Provide feedback**. Provide feedback

Amazon S3 > Buckets

Account snapshot
Storage lens provides visibility into storage usage and activity trends. Learn more
View Storage Lens dashboard

Buckets (1) Info
Buckets are containers for data stored in S3. Learn more

Copy ARN | Empty | Delete | Create bucket

Find buckets by name

Name	AWS Region	Access	Creation date
twitter-bucket-cs	US East (N. Virginia) us-east-1	Bucket and objects not public	December 3, 2022, 06:48:05 (UTC+05:30)

Feedback Looking for language selection? Find it in the new Unified Settings © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

Fig. 5. S3 Bucket Created

• *Creating IAM Roles and attaching AWS Policies:* We should create an AWS user and after that create and IAM role. Attach policies to that IAM Role.The policies that should be attached to IAM role are "AmazonKinesisFullAccess","kinesis:PutRecord","kinesis:PutRecords", "kinesis:GetShardIterator", "kinesis:GetRecords", "kinesis:DescribeStream". After that we should attach the role to EC2 instance and S3 Bucket also.

5.3 Connecting to Twitter and Streaming the Tweets

Our Second step is to connect to Twitter using the Twitter API. For this our requirement was Twitter developer account. we have made the Twitter developer account and got elevated access from Twitter to access their APIs.Using the bearer token we were able to connect with Twitter API. A python program named TwitterCalling.py using Tweepy library will run on an EC2 instance, fetching the tweets filtered with hashtags and keywords.We have put the filter ”Arts” and we can change the filter to ”Science”. We have run these streams many times by changing the filter value. The code for the data is provided in the repository.

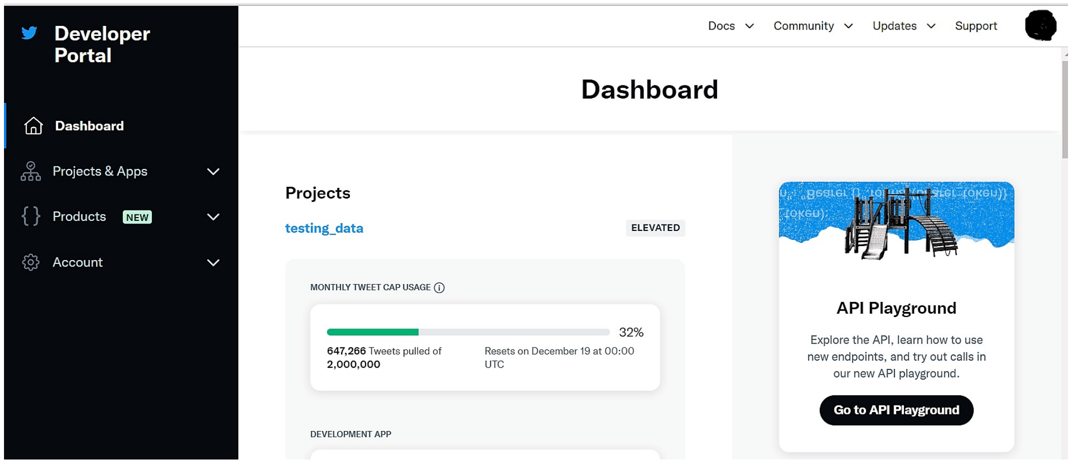


Fig. 6. Twitter Developer Portal

We have put the bearer token in the TwitterCalling.py file and with that, the connection with the Twitter API is built and after that, with Tweepy Library the tweets are fetched from Twitter and thereafter we send those tweets into the AWS Kinesis data stream.

5.4 Monitoring Kinesis Data Stream

Kinesis Data Stream digests the data in a managed, scalable way. Our next step is to Monitor the metrics and check if data goes into Kinesis.If the data goes into kinesis then the metrics will be made under the monitoring tab of Kinesis data stream.The below Fig 7 shows that the data is flowing into the Kinesis data stream.

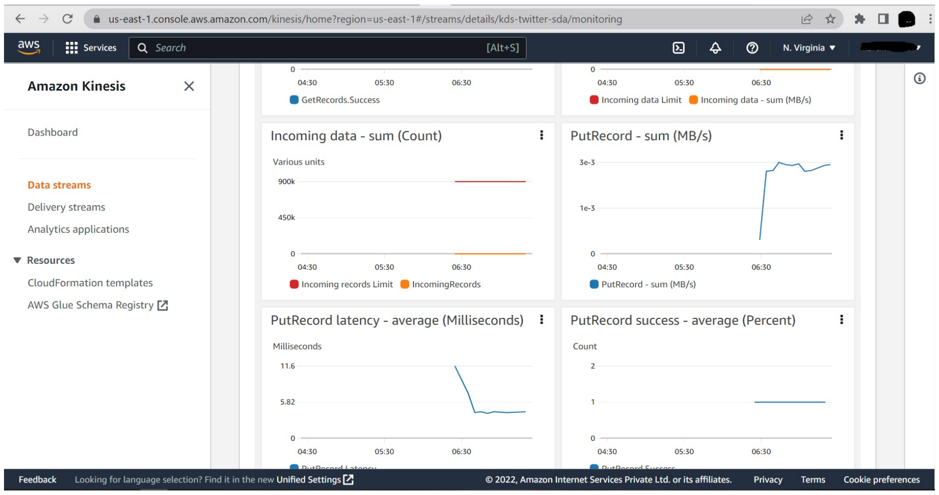
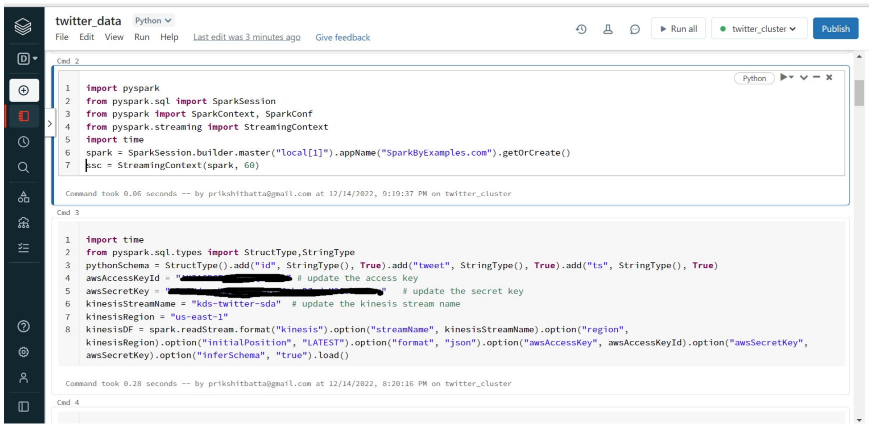


Fig. 7. Monitoring Kinesis

5.5 Connecting Data Stream to Databricks

Our next step is to get the data which is coming from the Data Stream. We have used Databricks community edition and attached a cluster to that and installed spark on that. With AWS credentials one can connect to an AWS account.We have made the data frame form that incoming data using spark and after that we are creating schema based on id,text and timestamp. The below figure shows some code of Databricks how to connect with the kinesis Data Stream.For the sake of security AWS credentials are not shown in the screenshot.

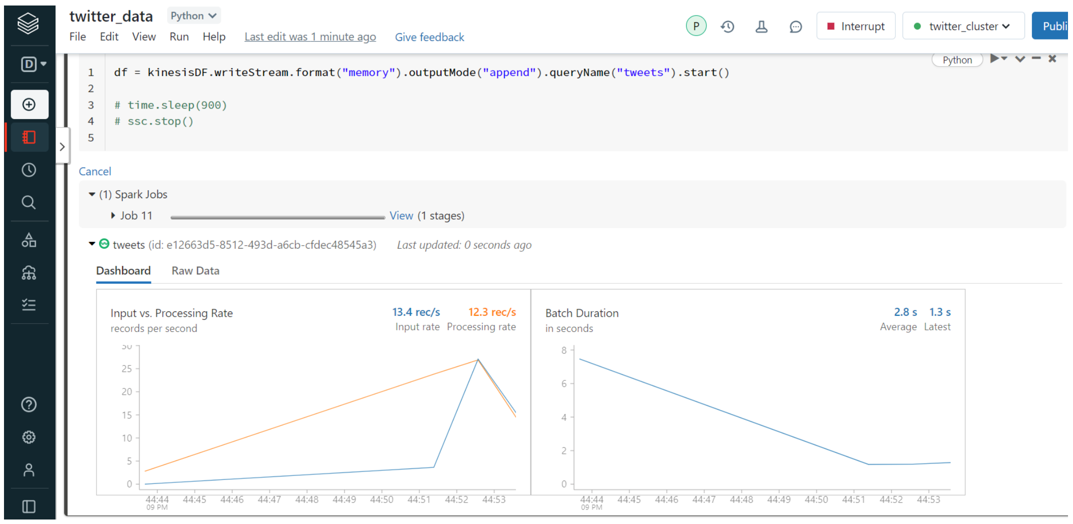


```
Cmd 2
import pyspark
from pyspark.sql import SparkSession
from pyspark import SparkContext, SparkConf
from pyspark.streaming import StreamingContext
import time
spark = SparkSession.builder.master("local[1]").appName("SparkByExamples.com").getOrCreate()
ssc = StreamingContext(spark, 60)

Command took 0.06 seconds -- by prikshitbatta@gmail.com at 12/14/2022, 9:19:37 PM on twitter_cluster
Cmd 3
import time
from pyspark.sql.types import StructType,StringType
pythonSchema = StructType().add("id", StringType(), True).add("tweet", StringType(), True).add("ts", StringType(), True)
awsAccessKeyId = "[redacted]" # update the access key
awsSecretKey = "[redacted]"   # update the secret key
kinesisStreamName = "kds-twitter-sda"  # update the kinesis stream name
kinesisRegion = "us-east-1"
kinesisDF = spark.readStream.format("kinesis").option("streamName", kinesisStreamName).option("region",
kinesisRegion).option("initialPosition", "LATEST").option("format", "json").option("awsAccessKey", awsAccessKeyId).option("awsSecretKey",
awsSecretKey).option("inferSchema", "true").load()

Command took 0.28 seconds -- by prikshitbatta@gmail.com at 12/14/2022, 8:20:16 PM on twitter_cluster
Cmd 4
```

Fig. 8. Connecting to Kinesis from Databricks

5.5 Converting the data to SQL Dataframes

After we have made the connection with the Kinesis data stream the data starts flowing and we can see that in the graph of tweets. Now our next step is to convert the received data into SQL data frames and then provide the data schema and apply transformations on it. The below figure shows the graph which tells that the data starts flowing into databricks.

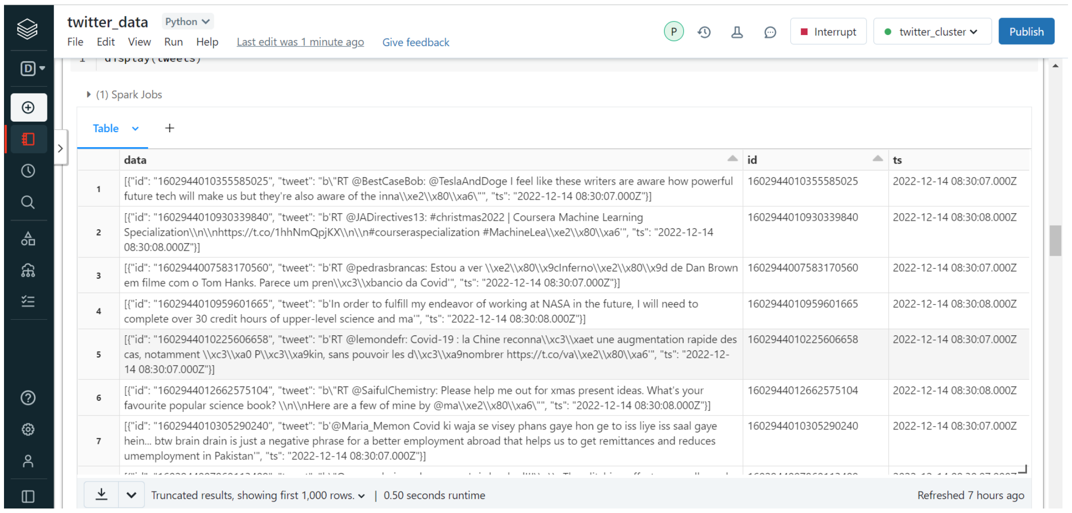


```
df = kinesisDF.writeStream.format("memory").outputMode("append").queryName("tweets").start()

# time.sleep(900)
# ssc.stop()

tweets (id: e12663d5-8512-493d-a6cb-cfdec48545a3)   Last updated: 0 seconds ago
```

Fig. 9. Data Flows into Databricks and is converted to SQL Dataframes

5.6 Applying User Defined Function to add columns and convert into id, Timestamp and Text

Now the next step is to convert the data frame to separate columns and we have converted one column to Id, Timestamp and tweet text from the same column that was there in data frame. The new data frame looks somewhat as shown in the below figure.



	data	id	ts
1	[{"id": "1602944010355585025", "tweet": "b\"RT @BestCaseBob: @TeslaAndDoge I feel like these writers are aware how powerful future tech will make us but they're also aware of the inna\\xe2\\x80\\xa6\"", "ts": "2022-12-14 08:30:07.000Z"}]	1602944010355585025	2022-12-14 08:30:07.000Z
2	[{"id": "1602944010930339840", "tweet": "b'RT @JADirectives13: #christmas2022 | Coursera Machine Learning Specialization\\n\\nhttps://t.co/1hhNmQpjKX\\n\\n#courseraspecialization #MachineLea\\xe2\\x80\\xa6'", "ts": "2022-12-14 08:30:08.000Z"}]	1602944010930339840	2022-12-14 08:30:08.000Z
3	[{"id": "1602944007583170560", "tweet": "b'RT @pedrasbrancas: Estou a ver \\xe2\\x80\\x9cInferno\\xe2\\x80\\x9d de Dan Brown em filme com o Tom Hanks. Parece um pren\\xc3\\xbancio da Covid'", "ts": "2022-12-14 08:30:07.000Z"}]	1602944007583170560	2022-12-14 08:30:07.000Z
4	[{"id": "1602944010959601665", "tweet": "b'In order to fulfill my endeavor of working at NASA in the future, I will need to complete over 30 credit hours of upper-level science and ma'", "ts": "2022-12-14 08:30:08.000Z"}]	1602944010959601665	2022-12-14 08:30:08.000Z
5	[{"id": "1602944010225606658", "tweet": "b'RT @lemondefr: Covid-19 : la Chine reconna\\xc3\\xaet une augmentation rapide des cas, notamment \\xc3\\xa0 P\\xc3\\xa9kin, sans pouvoir les d\\xc3\\xa9nombrer https://t.co/va\\xe2\\x80\\xa6'", "ts": "2022-12-14 08:30:07.000Z"}]	1602944010225606658	2022-12-14 08:30:07.000Z
6	[{"id": "1602944012662575104", "tweet": "b\"RT @SaifulChemistry: Please help me out for xmas present ideas. What's your favourite popular science book? \\n\\nHere are a few of mine by @ma\\xe2\\x80\\xa6\"", "ts": "2022-12-14 08:30:08.000Z"}]	1602944012662575104	2022-12-14 08:30:08.000Z
7	[{"id": "1602944010305290240", "tweet": "b'@Maria_Memon Covid ki waja se visey phans gaye hon ge to iss liye iss saal gaye hein... btw brain drain is just a negative phrase for a better employment abroad that helps us to get remittances and reduces umemployment in Pakistan'", "ts": "2022-12-14 08:30:07.000Z"}]	1602944010305290240	2022-12-14 08:30:07.000Z

Truncated results, showing first 1,000 rows. | 0.50 seconds runtime

Refreshed 7 hours ago

Fig. 10. New dataframe with Added Columns

5.7 Applying transformations and action on that data

Now we can apply any transformation and do sentimental analysis on this data after doing all this we will store the data in an S3 bucket and thereafter we will take data from S3 Bucket and Perform the Prediction of whether the data is "arts" data or "science" data using Machine Learning.

5.8 Storing the data into S3 bucket and Performing analysis on that data

Now we will store the data in S3 bucket and then we will download the CSV file from the S3 bucket.After that We will do the analysis of that particular data. We will so sentiment Analysis,Most Trending Hashtags, Most tagged Person, Hot words from this data. After this we will do some more data cleaning and store it to s3 for further analysis.

5.9 Applying Transformations and Storing to S3 for Machine Learning Analysis

After we have done the analysis on part one we will do some more data cleaning and transformation and store the data to S3 bucket and apply Ml model on that.

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. Naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For our purpose, we are going to use Bernoulli Naive Bayes. The parameters that we use to predict the class variable take up only values 1 or 0. If a word occurs in the tweets we will assign to 1 otherwise we will assign to 0.

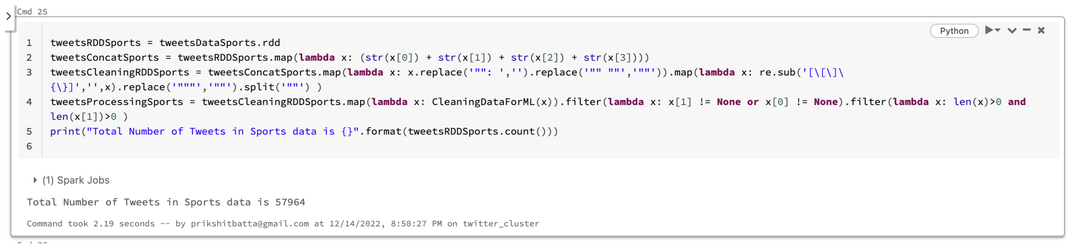
We use a model where a tweet text C depends on binary variables $A_1, A_2, A_3 \dots A_n$ which takes value 0 or 1. More specifically A_i denotes a word and if it is 1, then it is present in a tweet text. If it is 0, then it is not present in a tweet text. The parameters of this model are $\theta_c = P(C = 1)$, $\theta_{i1} = P(A_i = 1|C = 1)$, and $\theta_{i0} = P(A_i = 1|C = 0)$. Collectively, we will call $\theta_i = \{\theta_{i1}, \theta_{i0}\}$ and $\theta = \{\theta_c, \theta_1, \theta_2 \dots \theta_N\}$. To make predictions with this model, we compute $P(C = 1|a_1, a_2 \dots a_N) \infty P(a_1, a_2 \dots a_N|C = 1)P(C = 1) = \theta_c \Pi_{i=1}^{N} \theta_{i1}^{a_i}(1 - \theta_{i1})^{(1-a_i)}$. Similarly we compute $P(C = 0|a_1, a_2 \dots a_N) \infty (1 - \theta_c) \Pi_{i=1}^{N} \theta_{i0}^{a_i}(1 - \theta_{i0})^{(1-a_i)}$. In practice we compute log of both value and classify based of larger value, i.e. if $\log(P(C = 1|a_1, a_2 \dots a_N)) \geq \log(P(C = 0|a_1, a_2 \dots a_N))$, we classify the tweet text as 1.

We calculate θ using maximum likelihood estimation (MLE) and Laplace correction. Let the data is a set of tweet texts $d = \{d_1, d_2 \ldots d_M\}$ with total tweets $M = M_1 + M_0$ where M_1 is number tweets labeled as 1 and M_0 is number tweets labeled as 0. We define $d_j = \{a_{j1}, a_{j2} \ldots a_{jN}, c_j\}$, where N is total number of words in all tweets. Using MLE and Laplace correction we get $\theta_{i1} = \frac{\Sigma_{j|c_j=1} a_{ji}+1}{M_1+2}$ and $\theta_{i0} = \frac{\Sigma_{j|c_j=0} a_{ji}+1}{M_0+2}$.

We have arts.csv and science.csv files from the pipelines described above. The csv files contains 2102 and 3433 tweets respectively. In total, both files contains 13729 words. We used pySpark to create dictionaries with words as keys and values as list of lines.The code is included with the report.

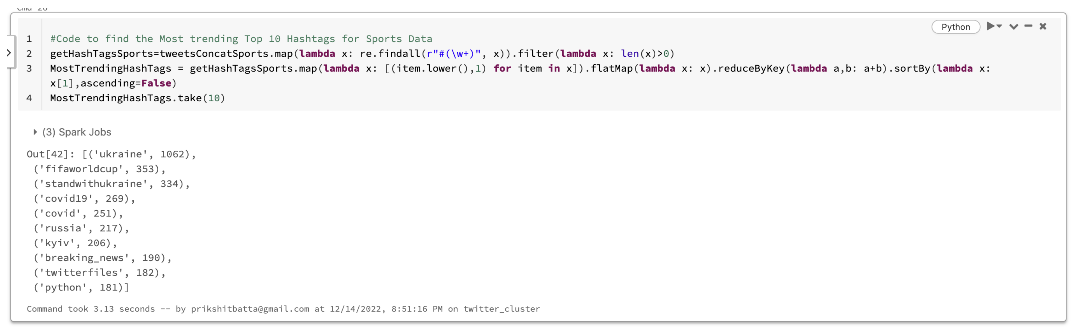
6 Results

6.1 Analysis of data stored to S3 in first Go:



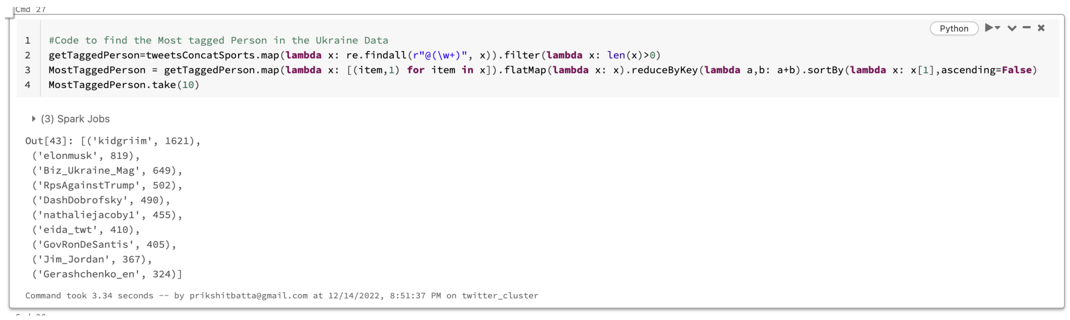
```
Cmd 25
tweetsRDDSports = tweetsDataSports.rdd
tweetsConcatSports = tweetsRDDSports.map(lambda x: (str(x[0]) + str(x[1]) + str(x[2]) + str(x[3])))
tweetsCleaningRDDSports = tweetsConcatSports.map(lambda x: x.replace('"": ','').replace('"" ""','""')).map(lambda x: re.sub('[\[\]\
{\}]','',x).replace('""""','""').split('""') )
tweetsProcessingSports = tweetsCleaningRDDSports.map(lambda x: CleaningDataForML(x)).filter(lambda x: x[1] != None or x[0] != None).filter(lambda x: len(x)>0 and
len(x[1])>0 )
print("Total Number of Tweets in Sports data is {}".format(tweetsRDDSports.count()))

(1) Spark Jobs
Total Number of Tweets in Sports data is 57964
Command took 2.19 seconds -- by prikshitbatta@gmail.com at 12/14/2022, 8:50:27 PM on twitter_cluster
```

Fig. 11. Total Tweets in the sports data



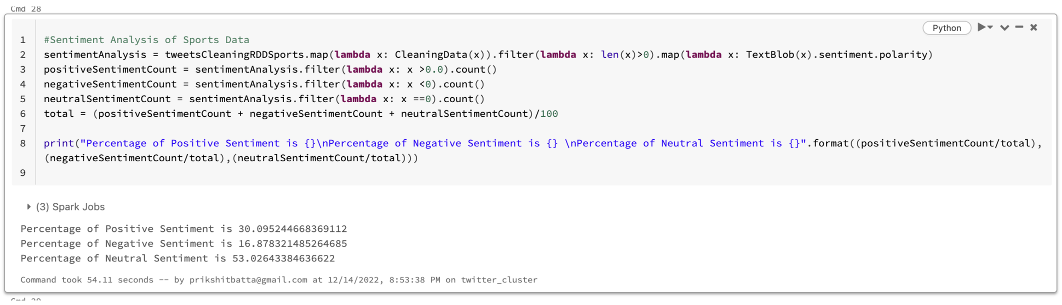
```
#Code to find the Most trending Top 10 Hashtags for Sports Data
getHashTagsSports=tweetsConcatSports.map(lambda x: re.findall(r"#(\w+)", x)).filter(lambda x: len(x)>0)
MostTrendingHashTags = getHashTagsSports.map(lambda x: [(item.lower(),1) for item in x]).flatMap(lambda x: x).reduceByKey(lambda a,b: a+b).sortBy(lambda x:
x[1],ascending=False)
MostTrendingHashTags.take(10)

(3) Spark Jobs
Out[42]: [('ukraine', 1062),
 ('fifaworldcup', 353),
 ('standwithukraine', 334),
 ('covid19', 269),
 ('covid', 251),
 ('russia', 217),
 ('kyiv', 206),
 ('breaking_news', 190),
 ('twitterfiles', 182),
 ('python', 181)]
Command took 3.13 seconds -- by prikshitbatta@gmail.com at 12/14/2022, 8:51:16 PM on twitter_cluster
```

Fig. 12. Most Trending Hashtags in sports



```
Cmd 27
#Code to find the Most tagged Person in the Ukraine Data
getTaggedPerson=tweetsConcatSports.map(lambda x: re.findall(r"@(\w+)", x)).filter(lambda x: len(x)>0)
MostTaggedPerson = getTaggedPerson.map(lambda x: [(item,1) for item in x]).flatMap(lambda x: x).reduceByKey(lambda a,b: a+b).sortBy(lambda x: x[1],ascending=False)
MostTaggedPerson.take(10)

(3) Spark Jobs
Out[43]: [('kidgriim', 1621),
 ('elonmusk', 819),
 ('Biz_Ukraine_Mag', 649),
 ('RpsAgainstTrump', 502),
 ('DashDobrofsky', 490),
 ('nathaliejacoby1', 455),
 ('eida_twt', 410),
 ('GovRonDeSantis', 405),
 ('Jim_Jordan', 367),
 ('Gerashchenko_en', 324)]
Command took 3.34 seconds -- by prikshitbatta@gmail.com at 12/14/2022, 8:51:37 PM on twitter_cluster
```

Fig. 13. Most Tagged Person in sports

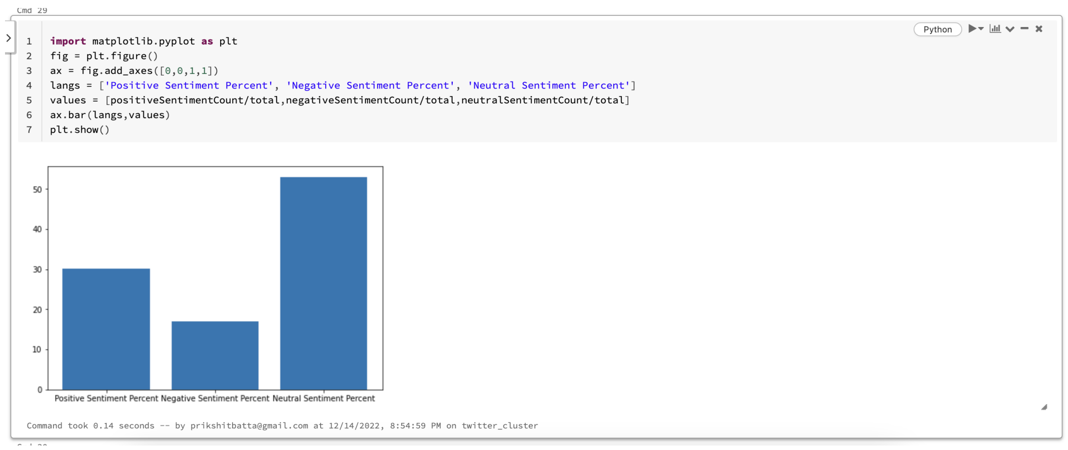


```
#Sentiment Analysis of Sports Data
sentimentAnalysis = tweetsCleaningRDDSports.map(lambda x: CleaningData(x)).filter(lambda x: len(x)>0).map(lambda x: TextBlob(x).sentiment.polarity)
positiveSentimentCount = sentimentAnalysis.filter(lambda x: x >0.0).count()
negativeSentimentCount = sentimentAnalysis.filter(lambda x: x <0).count()
neutralSentimentCount = sentimentAnalysis.filter(lambda x: x ==0).count()
total = (positiveSentimentCount + negativeSentimentCount + neutralSentimentCount)/100

print("Percentage of Positive Sentiment is {}\nPercentage of Negative Sentiment is {} \nPercentage of Neutral Sentiment is {}".format((positiveSentimentCount/total),
(negativeSentimentCount/total),(neutralSentimentCount/total)))
```

```
Percentage of Positive Sentiment is 30.095244668369112
Percentage of Negative Sentiment is 16.878321485264685
Percentage of Neutral Sentiment is 53.02643384636622
```

Command took 54.11 seconds -- by prikshitbatta@gmail.com at 12/14/2022, 8:53:38 PM on twitter_cluster

Fig. 14. Sentiment Analysis In Percentage in sports

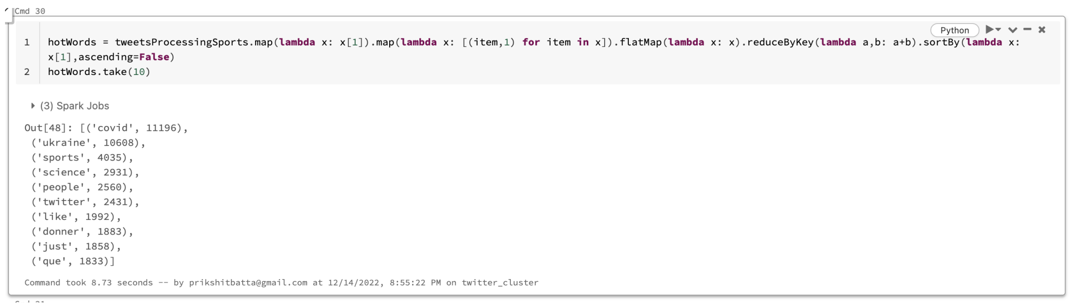


```
import matplotlib.pyplot as plt
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
langs = ['Positive Sentiment Percent', 'Negative Sentiment Percent', 'Neutral Sentiment Percent']
values = [positiveSentimentCount/total,negativeSentimentCount/total,neutralSentimentCount/total]
ax.bar(langs,values)
plt.show()
```

50
40
30
20
10
0
Positive Sentiment Percent
Negative Sentiment Percent
Neutral Sentiment Percent

Command took 0.14 seconds -- by prikshitbatta@gmail.com at 12/14/2022, 8:54:59 PM on twitter_cluster

Fig. 15. Sentiment Analysis Percentage Bargraph of sports



```
hotWords = tweetsProcessingSports.map(lambda x: x[1]).map(lambda x: [(item,1) for item in x]).flatMap(lambda x: x).reduceByKey(lambda a,b: a+b).sortBy(lambda x:
x[1],ascending=False)
hotWords.take(10)
```

```
Out[48]: [('covid', 11196),
 ('ukraine', 10608),
 ('sports', 4035),
 ('science', 2931),
 ('people', 2560),
 ('twitter', 2431),
 ('like', 1992),
 ('donner', 1883),
 ('just', 1858),
 ('que', 1833)]
```

Command took 8.73 seconds -- by prikshitbatta@gmail.com at 12/14/2022, 8:55:22 PM on twitter_cluster

Fig. 16. HotWords in the sports Tweets

Analysis of the Data, with various filters added to the streams of the data such as Ukraine, Sports, and Waterloo with number of tweets equal to 56635(27 mb), 57964(28.3 mb), 93843(48.2 mb) respectively:

1. It was seen that the Ukraine Hashtag was most used in most of the tweets across all the different data. It was analysed that the data with Sports and Waterloo contained Tweets with Hashtags for Ukraine and StandWithUkraine from time to time, As one of the Tweet said : "b'@joelandren @sullydish There you go again trying to deflect attention away from the war in Ukraine.'" As with FIFA World Cup the attention was deviating from the Ukraine and Covid to the Sports in the Sports data. Many People from time to time were pointing it out.

2. It was also noted that the second most used hashtags in Sports data was FifaWorldCup, after the Ukraine. Following the FIFA Hashtag was the stand with Ukraine, Covid19, Covid, Russia, Kyiv(Capital of Ukraine). This shows that for most of Twitter the Trending topic is Russia Vs Ukraine, Covid 19.

3. From our sentiment Analysis we can see that: Ukraine : Sports: Waterloo: Positive Sentiment is 32.11,

From Above Data we can say that the data with Ukraine has most negative percentage of negative sentiments followed by the Waterloo and Sports.

So the Sports does reduce the negative sentiments in human which can verified from the above Analysis. (Wasnt́ it obvious?)

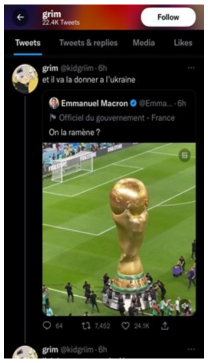
However the most Percentage of Neutral sentiments is in the Sports which can suggest that the People tweeting like to take the Sports for fun even if their country is participating or not(As in our case, India is not part of FIFA but we do follow it), thus providing them with feeling of Zen.(Pun intended here).

4. Then in our Analysis we can see the most tagged Person: Ukraine: elonmusk, rpsagainstrump Waterloo: elonmusk, rwmaloneMd, rpsagainstrump Sports: kidgriim, elonmusk

From the above results we can say that Elon Musk is getting more tagged in most of the tweets after accruing the Twitter. Elon Musk Support for Ukraine by giving Starlink stations to Ukraine and helping Ukrainians a lot, including the military.

The Kidgriim is most tagged person from the data has the person has 22.4k tweets in recent time for the FIFA. As shown below: @rwmaloneMd: is account for Robert W Malone, MD the Inventor and skeptic of mRNA vaccines. Who is a Scientist, Author, Speaker, Podcaster, Bioethicist.

For @ rpsAgainstTrump: Republicans against Trumpism have become active after many controversial accounts on twitter has been restored.



grim @kidgriim · 6h
et il va la donner a l'ukraine

Emmanuel Macron @Emma... · 6h
Officiel du gouvernement - France
On la ramène ?

64 7,452 24.1K

grim @kidgriim · 6h

Fig. 17. Kidgriim's Account Showing his Sports Activities

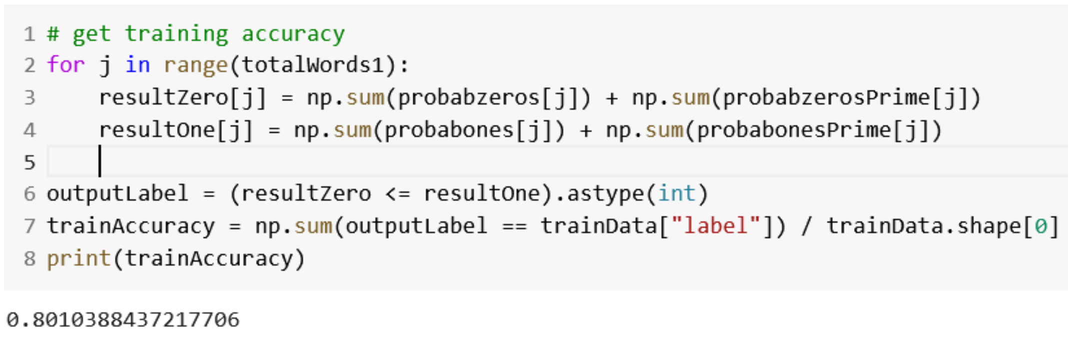
6.2 Perform Some transformation and Data Cleaning and Store to S3 for Machine Learning Analysis

We then split the data in 80% and 20% for training and testing purpose. Using training data we calculate the probabilities of each words being present given the subject of tweet is arts or science .

Then we classify the tweets method described in the Section 5.9. Using the data we got 80.1% training accuracy and 72.7% testing accuracy. So for training data our model correctly labeled 80.1% of tweets as Science or arts respectively. For 19.9% it predicts the opposite label or wrong label.

Train Tweets Arts	Train Tweets Science	Test Tweet Arts	Test Tweet Science	Total Words	Train Accuracy	Test Accuracy
2746	1681	687	421	13729	80.1	72.7

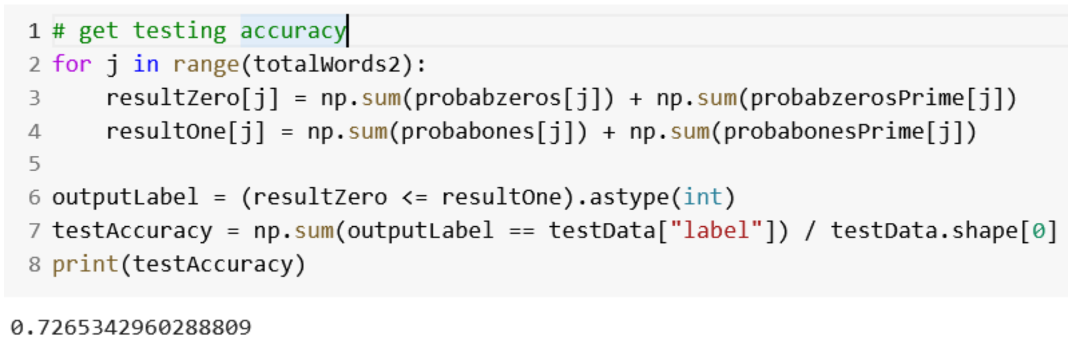
Table 1. Summary of Training and Test data and Accuracy



```
# get training accuracy
for j in range(totalWords1):
    resultZero[j] = np.sum(probabzeros[j]) + np.sum(probabzerosPrime[j])
    resultOne[j] = np.sum(probabones[j]) + np.sum(probabonesPrime[j])

outputLabel = (resultZero <= resultOne).astype(int)
trainAccuracy = np.sum(outputLabel == trainData["label"]) / trainData.shape[0]
print(trainAccuracy)
```

```
0.8010388437217706
```

Fig. 18. Training Accuracy



```
# get testing accuracy
for j in range(totalWords2):
    resultZero[j] = np.sum(probabzeros[j]) + np.sum(probabzerosPrime[j])
    resultOne[j] = np.sum(probabones[j]) + np.sum(probabonesPrime[j])

outputLabel = (resultZero <= resultOne).astype(int)
testAccuracy = np.sum(outputLabel == testData["label"]) / testData.shape[0]
print(testAccuracy)
```

```
0.7265342960288809
```

Fig. 19. Test Accuracy

7 Evaluation

We have done the analysis of real time Twitter data using AWS Kinesis and Databricks and then applied the machine learning model on this. As we can see that the accuracy of our machine learning model is 80.1% for training and 72.7% for testing data, we can say that our model is efficient. Thus we have performed a lot of analysis on the same live data received from Twitter. We can also scale this by running the live stream to several hours or days so that we can get the a large amount of data for better analysis. The Analysis done above are verified with the most recent tweets.

References

1. Assignment 0 CS 631

2. “PySpark documentation,” PySpark Documentation - PySpark 3.3.1 documentation. [Online]. Available: https://spark.apache.org/docs/latest/api/python/. [Accessed: 15-Dec-2022].

3. “PySpark archives,” Spark by Examples, 13-Dec-2022. [Online]. Available: https://sparkbyexamples.com/pyspark. [Accessed: 15-Dec-2022].

4. P. Gupta, “Spark Streaming for Beginners,” Medium, 16-Oct-2020. [Online]. Available: https://towardsdatascience.com/spark-streaming-for-beginners-a0170113e479. [Accessed: 15-Dec-2022].