Modern Data Architectures Twitter Data Stream Processing

GROUP F
Celia Assmuth Oreja, Mikael Dzhaneryan, Aditya Gavali Sunil, Alexandra Mathay, Paul Ribes, Fernando Suárez Pinto, Eric von Stockar

Contents

Business Value

Real-time customer sentiment analysis on a social media channel such as Twitter in which the business can quickly react to negative user sentiments, analyse the data, and make business decisions.

Data Pipeline

SOURCE

Twitter

The data source will be from Tweets that mention the word "Uber" on Twitter

INGESTION

Kafka

Streaming data will be ingested through Apache Kafka

STORAGE

Kafka

Streaming data is stored in a Kafka topic

PROCESSING

Spark

Stream processing sentiment analysis will be performed to the data

SERVING

MariaDB

The serivng layer will be in MariaDB where the business team can query the Twitter data

BI TOOL

Tableau

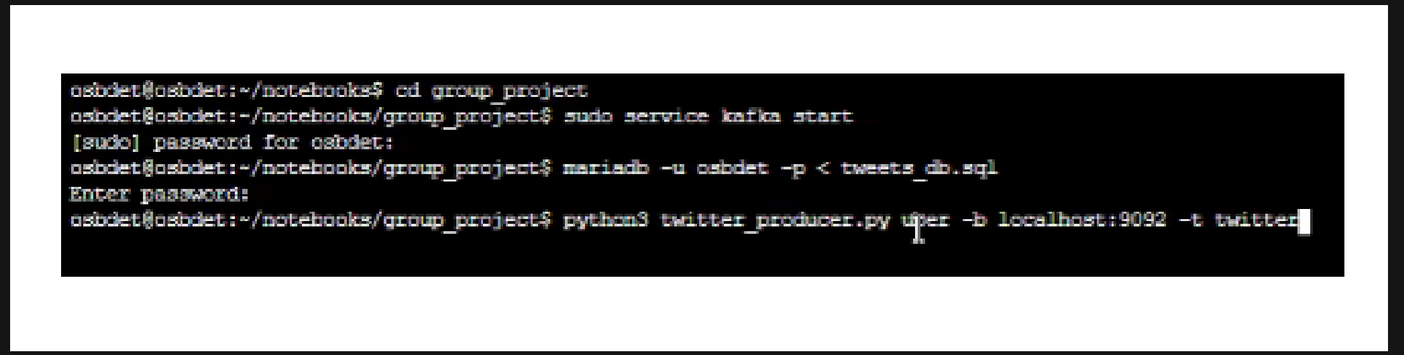
BI Tools can connect to the serving layer to create visualisations that aide in decision-making.

Data Sources

Twitter

Keywords: UBER

Stream processing



DATA INGESTION AND STORAGE

INGESTION VIA TWEEPY

Obtain data from Twitter

KAFKA PRODUCER

Using KAFKA we ingested streaming tweets containing the word "UBER"

KAFKA TOPIC

Once ingested we store the raw twitter data into a kafka topic

DATA PROCESSING

PYSPARK

We used the python interface for Apache Spark to process our data.

DATA CLEANING

We removed retweets, handles, hashtags, URLs, punctuation from the text. Transformed all words to lowercase.

```
.withColumn("lang",col("fields").getItem(40).cast(StringType())) \
.withColumn("text_no_rt", F.regexp_replace("text", r'RT @\w+: ', " ").cast(StringType())) \
.withColumn("text_no_handle", F.regexp_replace("text_no_rt", r'(@[A-Za-z0-9_-]+)', " ").cast(StringType())) \
.withColumn("text_no_hashtags", F.regexp_replace("text_no_handle", r'(#[A-Za-z0-9_-]+)', " ").cast(StringType())
.withColumn("text_no_urls", F.regexp_replace("text_no_hashtags", r'(\w+:\/\/\S+)', " ").cast(StringType())) \
.withColumn("text_no_punct", F.regexp_replace("text_no_urls", r'[\.\,\|\?\:\;\-\=\"]', " ").cast(StringType()))
.withColumn('text_lower', lower(col('text_no_punct')).cast(StringType())) \
.drop("fields")
```

```
In [5]: enohlcvDF = ohlcvDF.where((col('lang').like('en')))
```

SENTIMENT ANALYSIS

We used the algorithms NKLT & VADER to classify the sentiments of the twitter users.

```
In [6]: from nltk.sentiment import vader
        from vaderSentiment import vaderSentiment

        nltk = vader.SentimentIntensityAnalyzer()
        vader = vaderSentiment.SentimentIntensityAnalyzer()

In [7]: from pyspark.sql.functions import udf
        from pyspark.sql.types import MapType, StringType, FloatType

        nltk_polarityUDF = udf(lambda z: nltk.polarity_scores(z), MapType(StringType(), FloatType()))
        vaderS_polarityUDF = udf(lambda z: vader.polarity_scores(z), MapType(StringType(), FloatType()))

        enohlcvDF = enohlcvDF.withColumn('nltk', nltk_polarityUDF(enohlcvDF['text_lower'])['compound'])
        enohlcvDF = enohlcvDF.withColumn('vader', vaderS_polarityUDF(enohlcvDF['text_lower'])['compound'])
        enohlcvDF
```

DATA SERVING

MariaDB

CREATE DATABASE AND TABLE

In MariaDB we created the database and the table schema where the data was saved as a serving layer

```
osbdet@osbdet:~/notebooks$ cd group_project
osbdet@osbdet:~/notebooks/group_project$ sudo service kafka start
[sudo] password for osbdet:
osbdet@osbdet:~/notebooks/group_project$ mariadb -u osbdet -p < tweets_db.sql
Enter password:
osbdet@osbdet:~/notebooks/group_project$
```

SEND DATA FROM SPARK

After the processing we send the data in micro-batches

```
def foreach_batch_function(df, epoch_id):
    print ("Batch %d received" % epoch_id)

    # databases connection properties
    url = "jdbc:mariadb://localhost:3306/twitter"
    table = "Tweets"
    mode = "append"
    props = {"user": "osbdet",
             "password":"osbdet123$",
             "driver":"org.mariadb.jdbc.Driver"}
    (df.select('created_at',
               'id',
               'id_str',
               'text',
               'source',
               'truncated',
               'in_reply_to_status_id',
               'in_reply_to_user_id'
```

```
query = (enohlcvDF.writeStream.foreachBatch(lambda df,epochId:foreach_batch_function(df, epochId))).start()
```

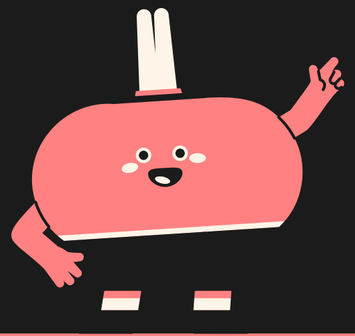
Analysis Results

SENTIMENT ANALYSIS

Happy

Calm
Happy
I'm focused
Feeling okay
In control

Neutral



Unhappy

Angry
Scared
Panic
I want to yell
I'm not in control

SENTIMENT ANALYSIS

Some Examples:

```
In [24]: print(ohlcvDF.toPandas().iloc[121]['text_lower'])
         print(ohlcvDF.toPandas().iloc[121][['nltk','vader']])

         i absolutely love uber  their drivers are the best  i am in love with the service the driver was so hot
         nltk      0.9347000000000000000000000000000000000000
         vader     0.9312000000000000000000000000000000000000
         Name: 121, dtype: object
```

```
25]: print(ohlcvDF.toPandas().iloc[213]['text_lower'])
     print(ohlcvDF.toPandas().iloc[213][['nltk','vader']])

     i hate uber
     nltk     -0.5719000000000000000000000000000000000000
     vader    -0.5719000000000000000000000000000000000000
     Name: 213, dtype: object
```

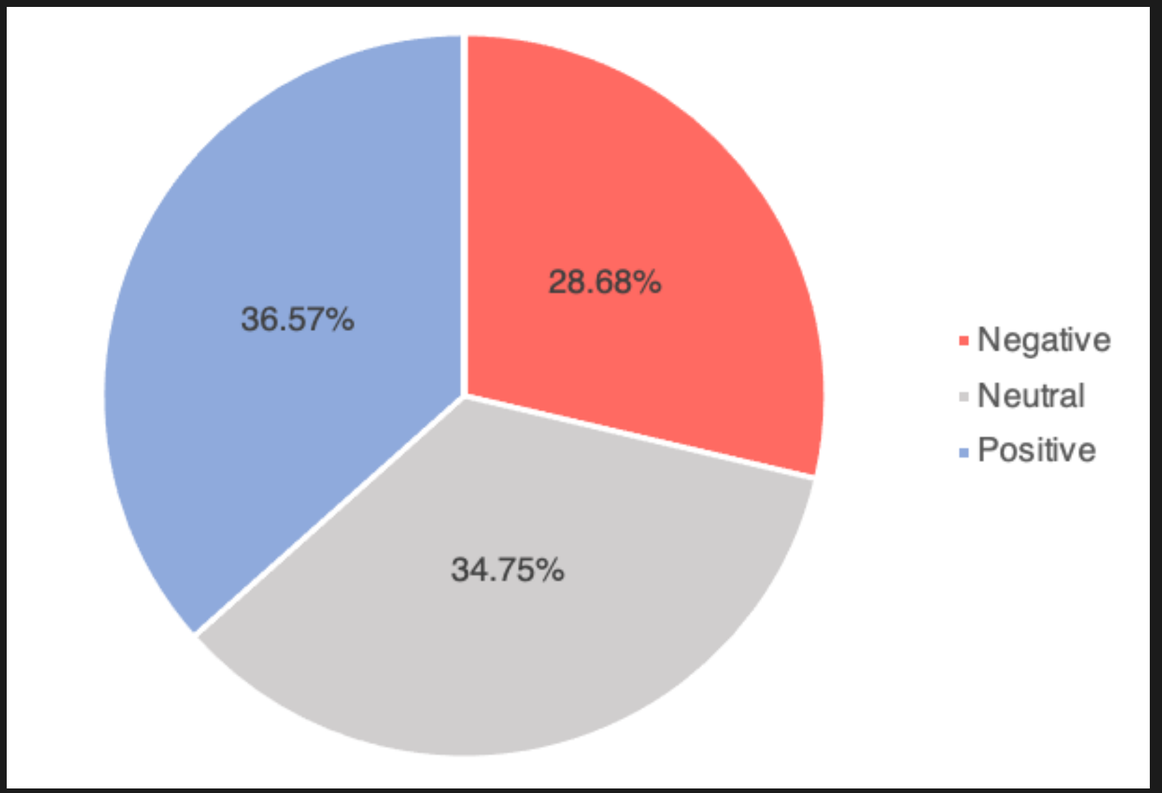
SENTIMENT ANALYSIS

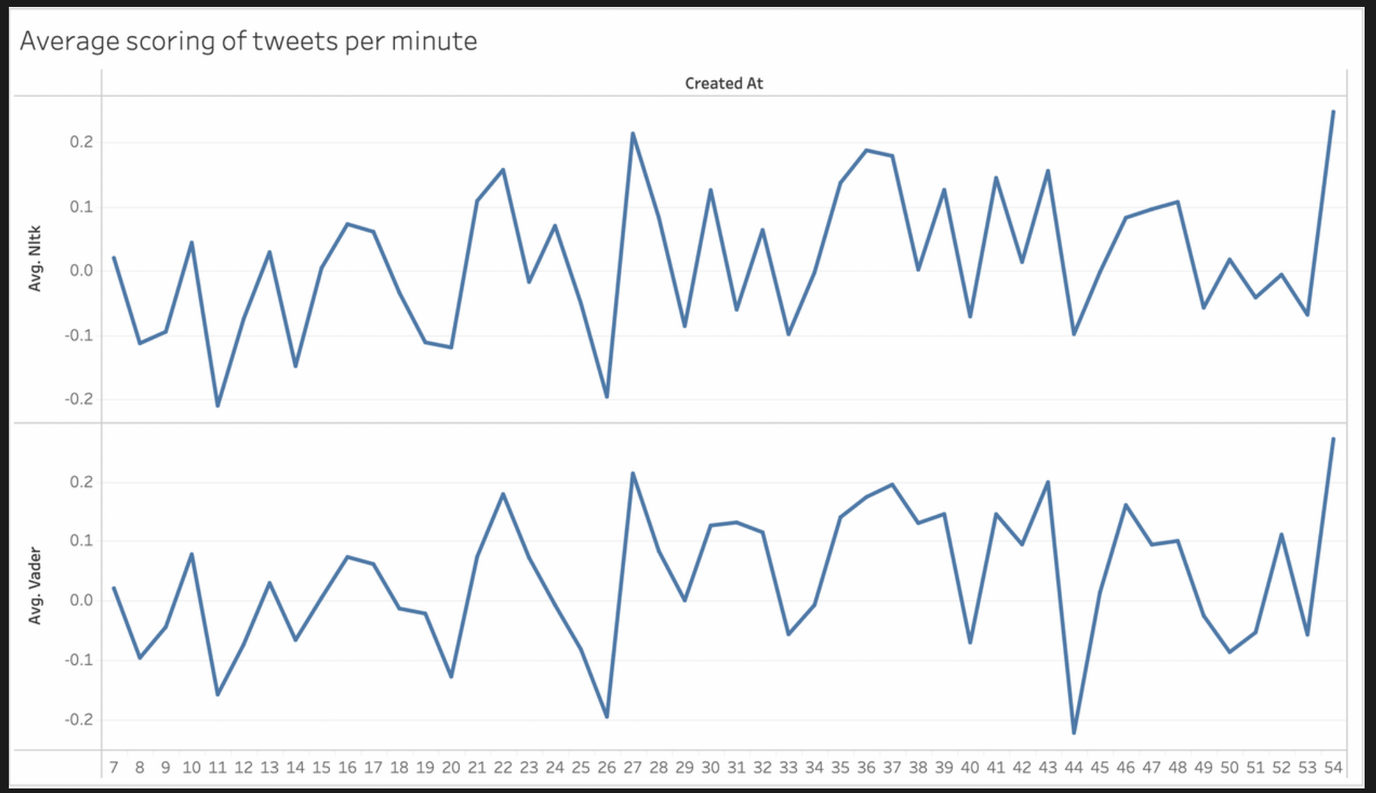
Nltk vs Vader

Abc mariadb.csv **Text No Punct**	Abc mariadb.csv **Text Lower**	# mariadb.csv **Nltk**	# mariadb.csv **Vader**
I think my Uber driver noticed ... but I already knew he was down 😈	i think my uber driver notic...	0.00000	0.61240
Are you saying Ola Uber	are you saying ola uber	0.00000	0.00000
I had the worst Uber /taxi experience	i had the worst uber /taxi ...	-0.62490	-0.62490

VISUALISATIONS

Having a look at the tweets scoring





Key Takeaways

Creating a Kappa Architecture Pipeline

We have created Kappa Architecture Pipeline, where there are two main layers, the streaming, and the serving layer. For use cases where the business needs to react quickly, this is a great way to provide the needed data to users.

Stream Processing using Kafka and Spark

Kafka and Spark are great tools for building a stream processing pipeline

“If you make customers unhappy in the physical world, they might each tell 6 friends. If you make customers unhappy on the Internet, they can each tell 6,000 friends.”

Jeff Bezos