Real Time Twitter Streaming

Bhargav Pabbisetti

Introduction

High Level Project Encapsulation

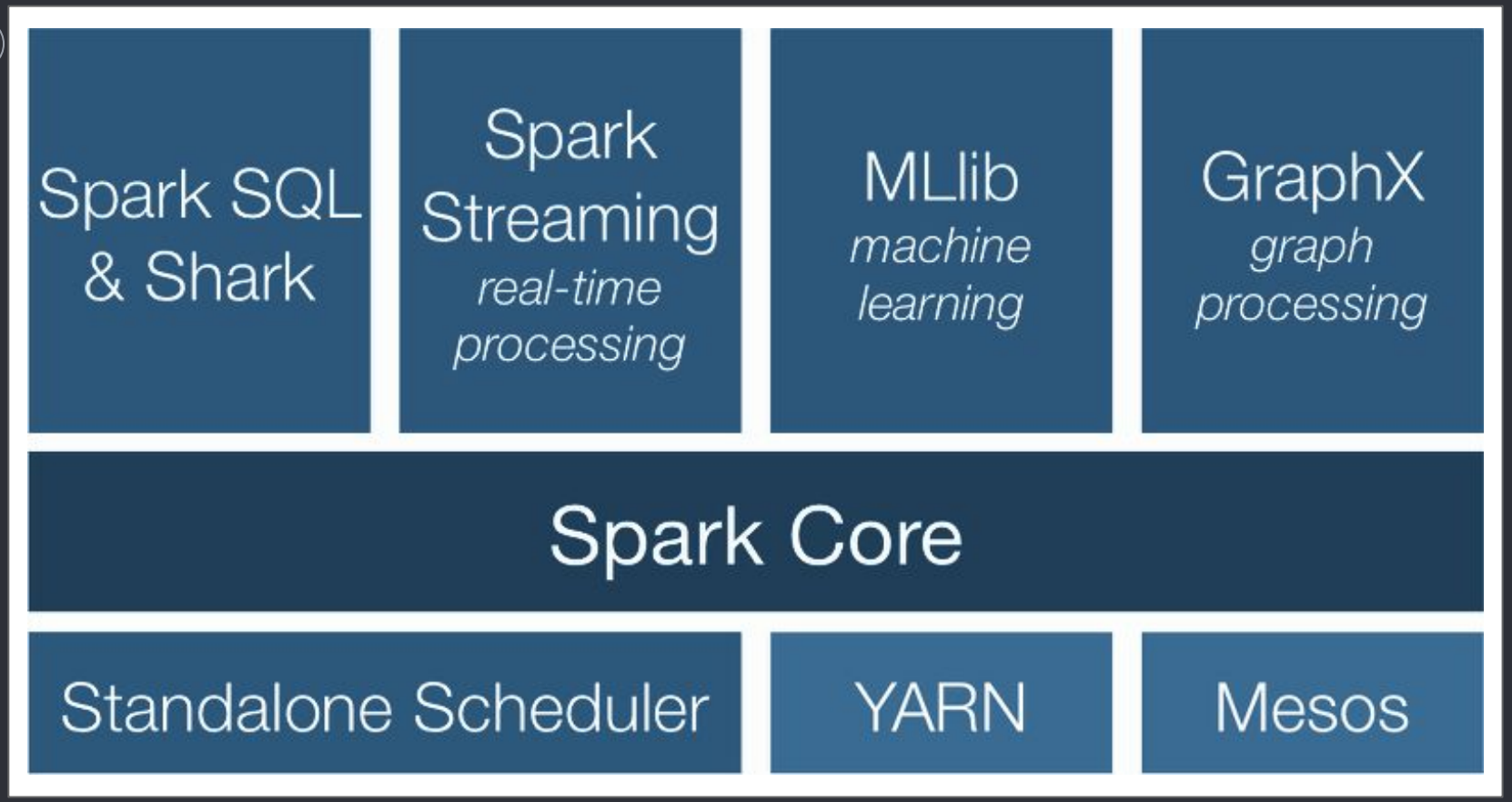
- Distributed Stream Processing » Apache Spark
- Machine Learning » Naive Bayes Classifier [Apache Spark MLlib implementation]
- Publish data to Redis
- Visualization » Sentiment visualization on a World map using Datamaps

What is Spark and why we should use it?

Spark is a lightning fast cluster computing platform

- Spark lets you run programs upto 100x faster in memory and 10x faster on disk than Hadoop.
- Spark is ideal for iterative, interactive and event stream processing.
- We can take advantage of many additional libraries that are built on top of Spark Core.

Spark Ecosystem



Resilient Distributed Dataset(RDD)

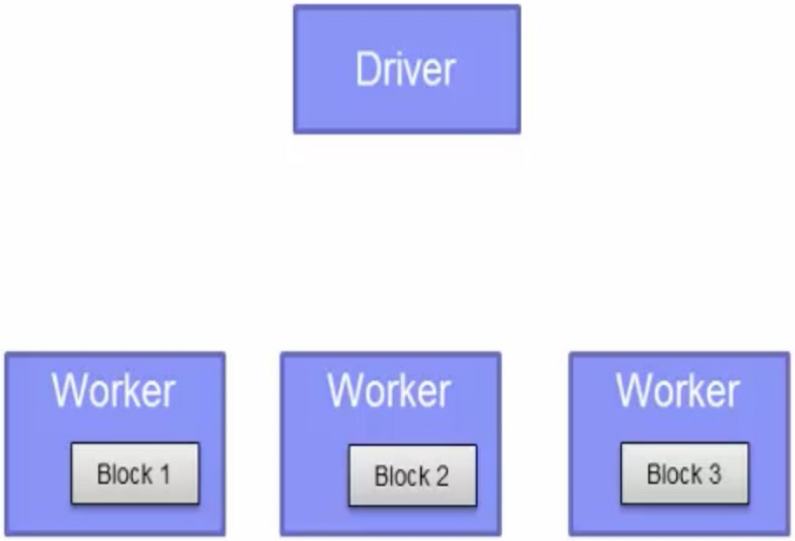
- Spark’s primary abstraction
- It is a collection of elements that is partitioned across a cluster and it can be operated on in parallel.
- Immutable
- Two types of RDD operations
  - Transformations
    - Creates Direct Acyclic Graph(DAG)
    - Lazy evaluation
    - No return value
  - Actions
    - Performs the transformations and the action that follows
    - Returns a value
- Fault tolerance
- Caching

Shared Variables

- Two types of variables
  - Broadcast variables
    - Read only copy on each machine
    - Distribute broadcast variable using efficient broadcast algorithm
  - Accumulators
    - Variables added through an associative operation
    - Implement counters and sums
    - Only the driver can read the accumulator

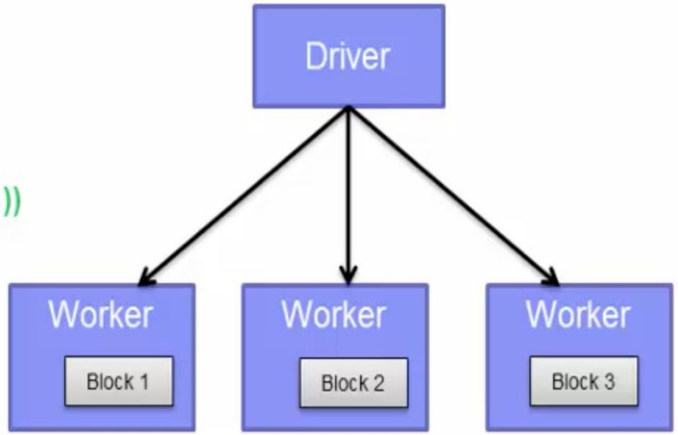
Basic Code Execution

```
// Creating the RDD
val logFile = sc.textFile("hdfs://...")
// Transformations
val errors = logFile.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))
//Caching
messages.cache()
// Actions
messages.filter(_.contains("mysql")).count()
messages.filter(_.contains("php")).count()
```



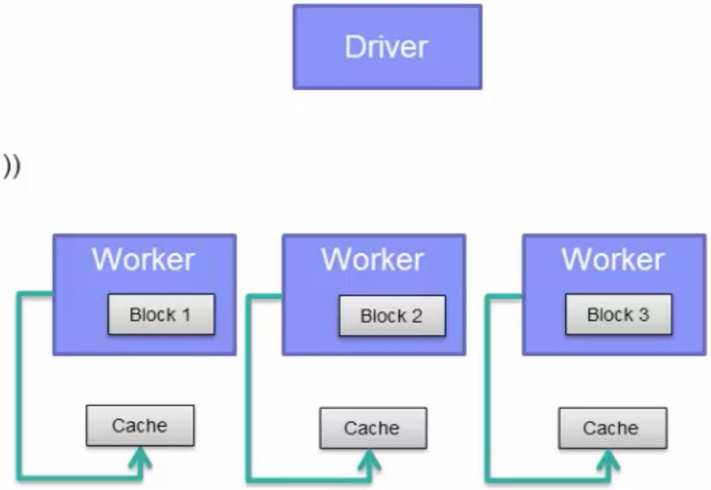
The data is partitioned into different blocks

```
// Creating the RDD
val logFile = sc.textFile("hdfs://...")
// Transformations
val errors = logFile.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))
//Cache
messages.cache()
// Actions
messages.filter(_.contains("mysql")).count()
messages.filter(_.contains("php")).count()
```



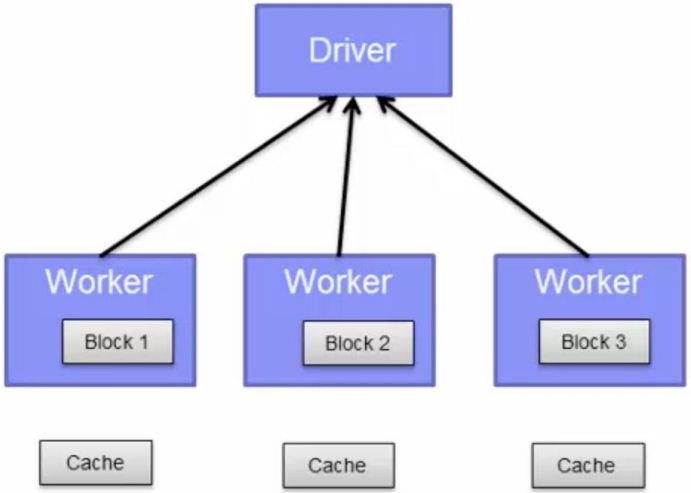
Driver sends the code to be executed on each block

```
// Creating the RDD
val logFile = sc.textFile("hdfs://...")
// Transformations
val errors = logFile.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))
//Caching
messages.cache()
// Actions
messages.filter(_.contains("mysql")).count()
messages.filter(_.contains("php")).count()
```



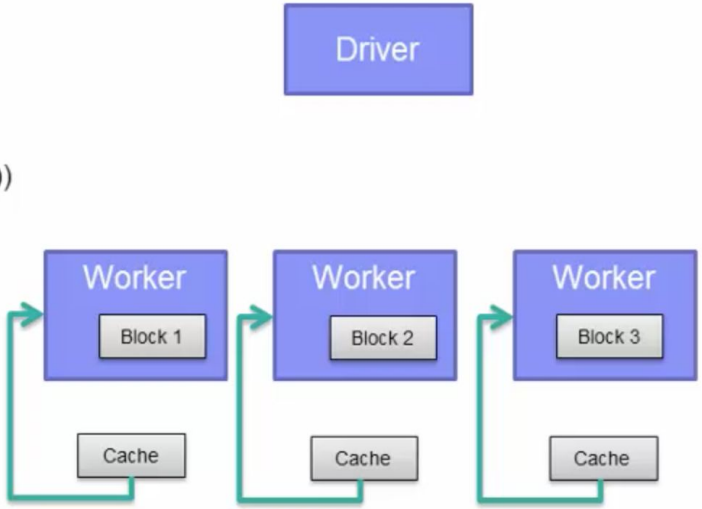
Process + cache data

```
// Creating the RDD
val logFile = sc.textFile("hdfs://...")
// Transformations
val errors = logFile.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))
//Caching
messages.cache()
// Actions
messages.filter(_.contains("mysql")).count()
messages.filter(_.contains("php")).count()
```

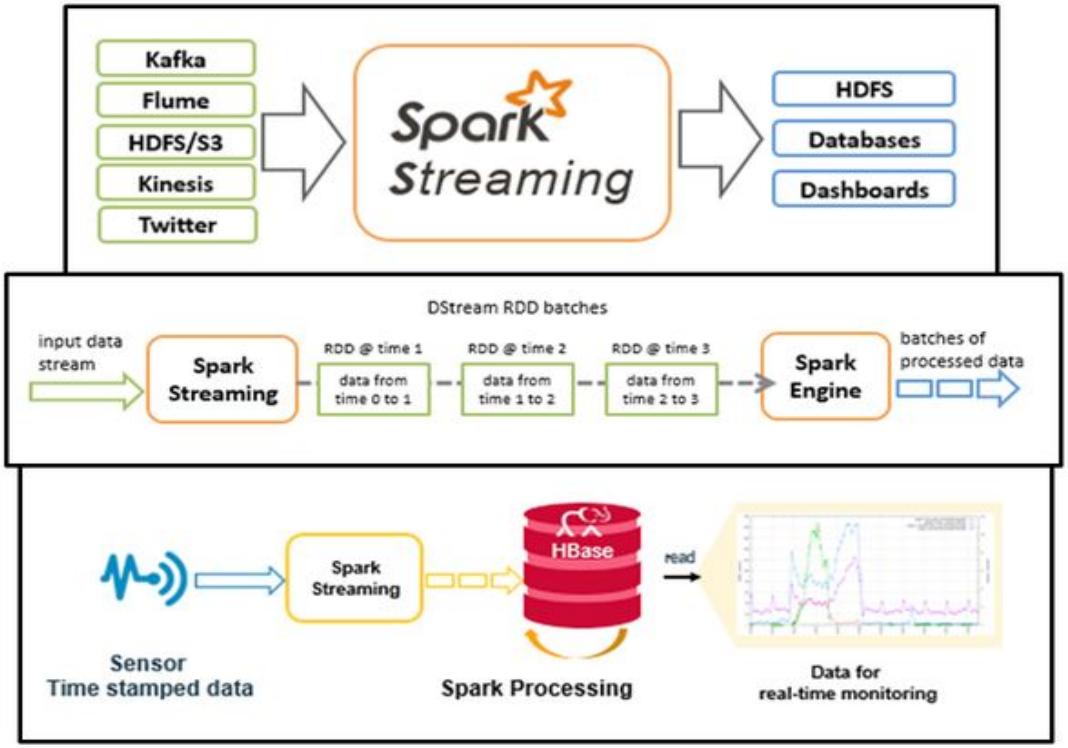


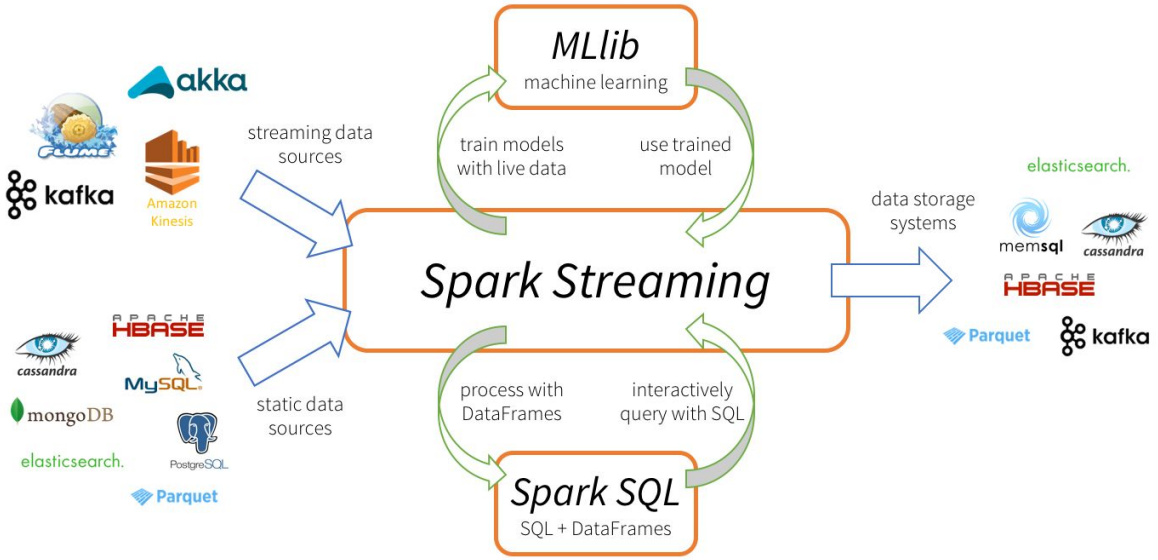
Send the data back to the driver

```
// Creating the RDD
val logFile = sc.textFile("hdfs://...")
// Transformations
val errors = logFile.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))
//Caching
messages.cache()
// Actions
messages.filter(_.contains("mysql")).count()
messages.filter(_.contains("php")).count()
```



Process from cache


Kafka
Flume
HDFS/S3
Kinesis
Twitter
Spark Streaming
HDFS
Databases
Dashboards
DStream RDD batches
input data stream
Spark Streaming
RDD @ time 1
data from time 0 to 1
RDD @ time 2
data from time 1 to 2
RDD @ time 3
data from time 2 to 3
Spark Engine
batches of processed data
Spark Streaming
HBase
read
Sensor
Time stamped data
Spark Processing
Data for
real-time monitoring


MLlib
machine learning
train models with live data
use trained model
akka
Flume
kafka
Amazon Kinesis
streaming data sources
Spark Streaming
data storage systems
elasticsearch.
memsql
cassandra
HBASE
Parquet
kafka
cassandra
HBASE
MySQL
mongoDB
elasticsearch.
PostgreSQL
Parquet
static data sources
process with DataFrames
interactively query with SQL
Spark SQL
SQL + DataFrames

Sample code using twitter4j

```
val oAuth: Some[OAuthAuthorization] = Authorization.TwitterAuth()
val rawTweets = TwitterUtils.createStream(ssc, oAuth)  //twitter input dstreams
```

```
val classifiedTweets = rawTweets.filter(hasGeoLocation)
  .map(Sentimentprediction)
```

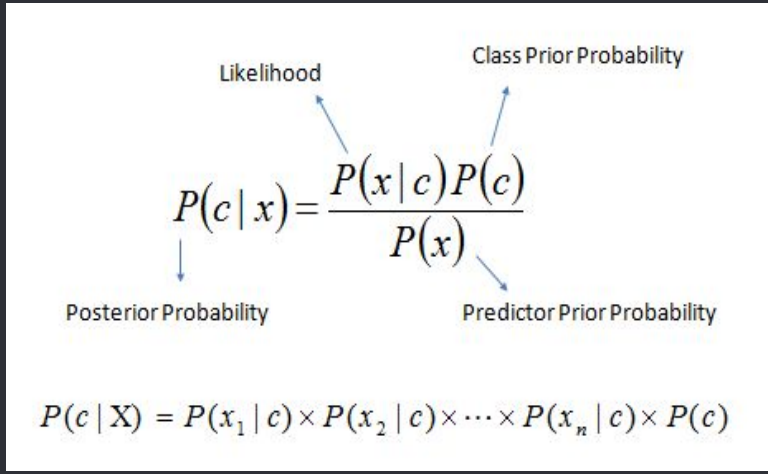
```
/**
 * Predicts the sentiment of the tweet passed.
 * Invokes Stanford Core NLP and MLlib methods for identifying the tweet sentiment.
 *
 * @param status -- twitter4j.Status object.
 * @return tuple with Tweet ID, Tweet Text, Core NLP Polarity, MLlib Polarity, Latitude, Longitude, Profile Image URL, Tweet Date.
 */
def Sentimentprediction(status: Status): (Long, String, String, Int, Int, Double, Double, String, String) = {
  val tweetText = replaceNewLines(status.getText)

  val (corenlpSentiment, mllibSentiment) = {
    // If tweet is in English, compute the sentiment by MLlib and also with Stanford CoreNLP.
    if (isTweetInEnglish(status)) {
      (CoreNLPSentimentAnalyzer.computeWeightedSentiment(tweetText),
        MLlibHelper.computeSentiment(tweetText, stopWordsList, naiveBayesModel))
    } else {
      // TODO: all non-English tweets are defaulted to neutral.
      // TODO: this is a workaround :: as we cant compute the sentiment of non-English tweets with our current model.
      (0, 0)
    }
  }
  (status.getId,
    status.getUser.getScreenName,
    tweetText,
    corenlpSentiment,
    mllibSentiment,
    status.getGeoLocation.getLatitude,
    status.getGeoLocation.getLongitude,
    status.getUser.getOriginalProfileImageURL,
    simpleDateFormat.format(status.getCreatedAt))

}
```

Spark MLlib

- Multinomial Naive Bayes is used to train 1.6 million tweets collected by Stanford University.
- Data is in csv format with 6 fields
  - 0 - the polarity of the tweet(0=negative, 2=neutral, 4=positive)
  - 1 - the id of the tweet
  - 2 - the date of the tweet
  - 3 - the user that tweeted
  - 4 - the text of the tweet
- Term Frequency are the features
- Achieved 78% accuracy.
- Pre trained model from stanford was also used to determine sentiment of a text
  - This model uses Recursive Neural Network that builds on top of grammatical structures



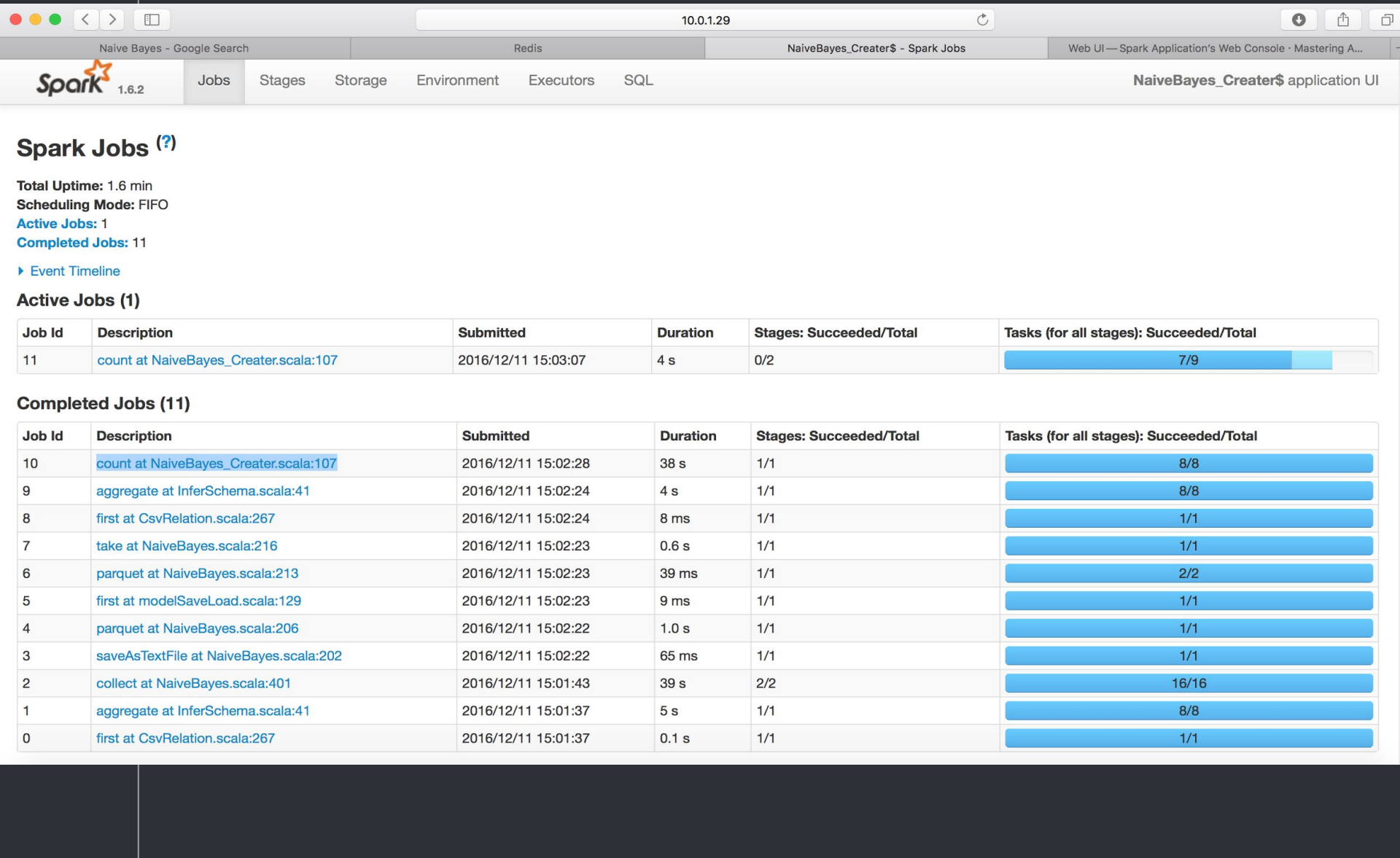
MLlib Code Snapshots

```scala
/**
 * Creates a Naive Bayes Model of Tweet and its Sentiment from the Sentiment140 file.
 *
 * @param sc            -- Spark Context.
 * @param stopWordsList -- Broadcast variable for list of stop words to be removed from the tweets.
 */
def createAndSaveNBModel(sc: SparkContext, stopWordsList: Broadcast[List[String]]): Unit = {
  val trainDF = loadTrainingFile(sc, PropertiesLoader.SENTIMENT140_TRAIN_DATA_PATH)

  val labeled_rdd = trainDF.select("POLARITY", "STATUS").rdd.map{
    case Row(polarity: Int, tweet: String) =>
      val tweetInWords: Seq[String] = MLlibHelper.cleanTweets(tweet, stopWordsList.value)
      LabeledPoint(polarity, MLlibHelper.transformFeatures(tweetInWords))
  }
  labeled_rdd.cache()

  val naiveBayesModel: NaiveBayesModel = NaiveBayes.train(labeled_rdd, lambda = 1.0, modelType = "multinomial")
  naiveBayesModel.save(sc, PropertiesLoader.NAIVEBAYES_MODEL_PATH)
}
```

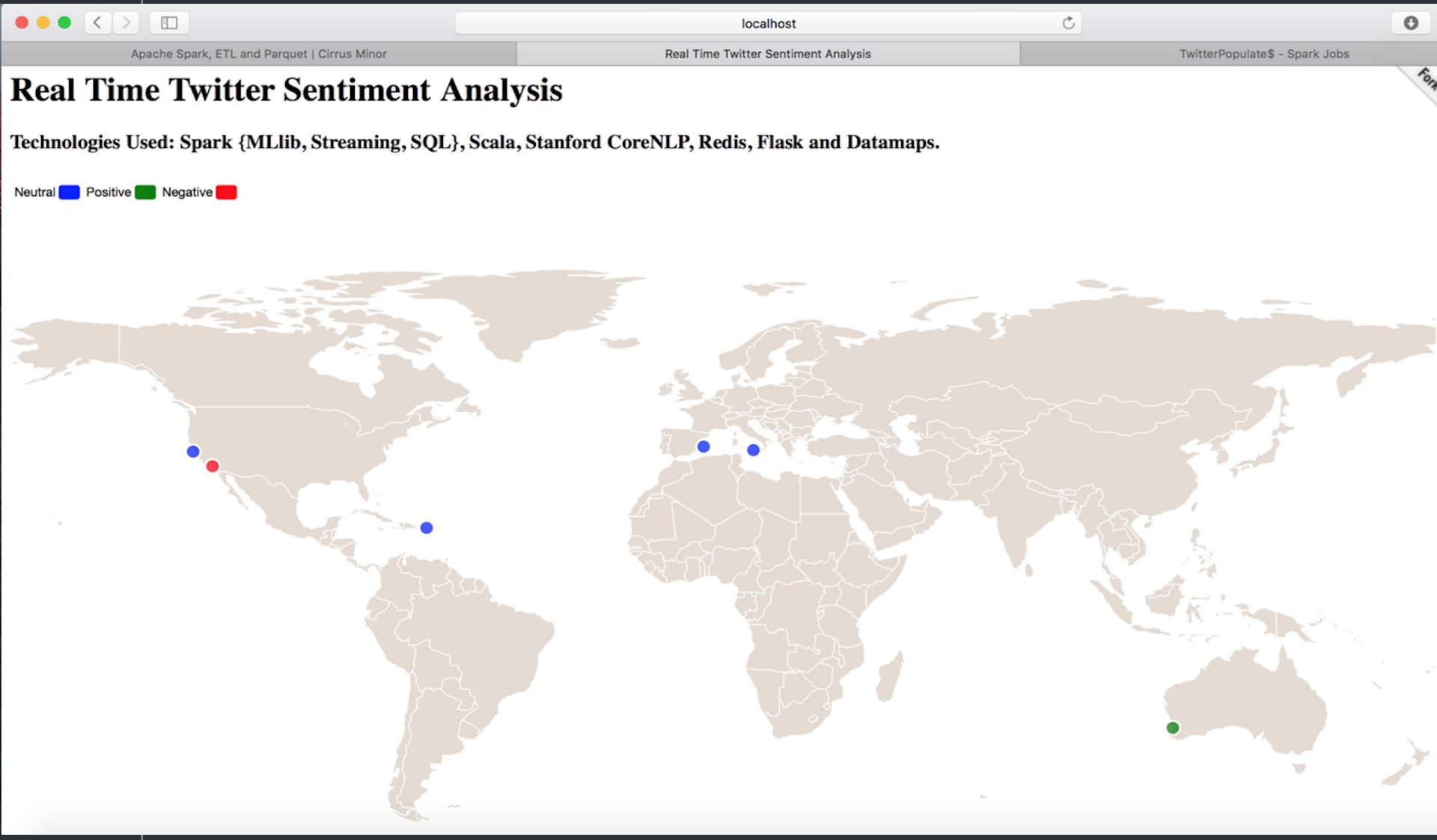
Spark UI Mllib



Web Server and Visualization

- Messaging System is built using Redis.
- Result of tweets are published to Redis which is subscribed by front end webapp.
- Web application uses server sent event.
- Python Flask web server
- HTML5, Javascript, D3 js are used to visualize real time tweets
- World map is created using datamaps.
- Bubbles represent live tweet.
- Hovering over a bubble reveals details of the tweet

Snapshots of Real Time Visualization



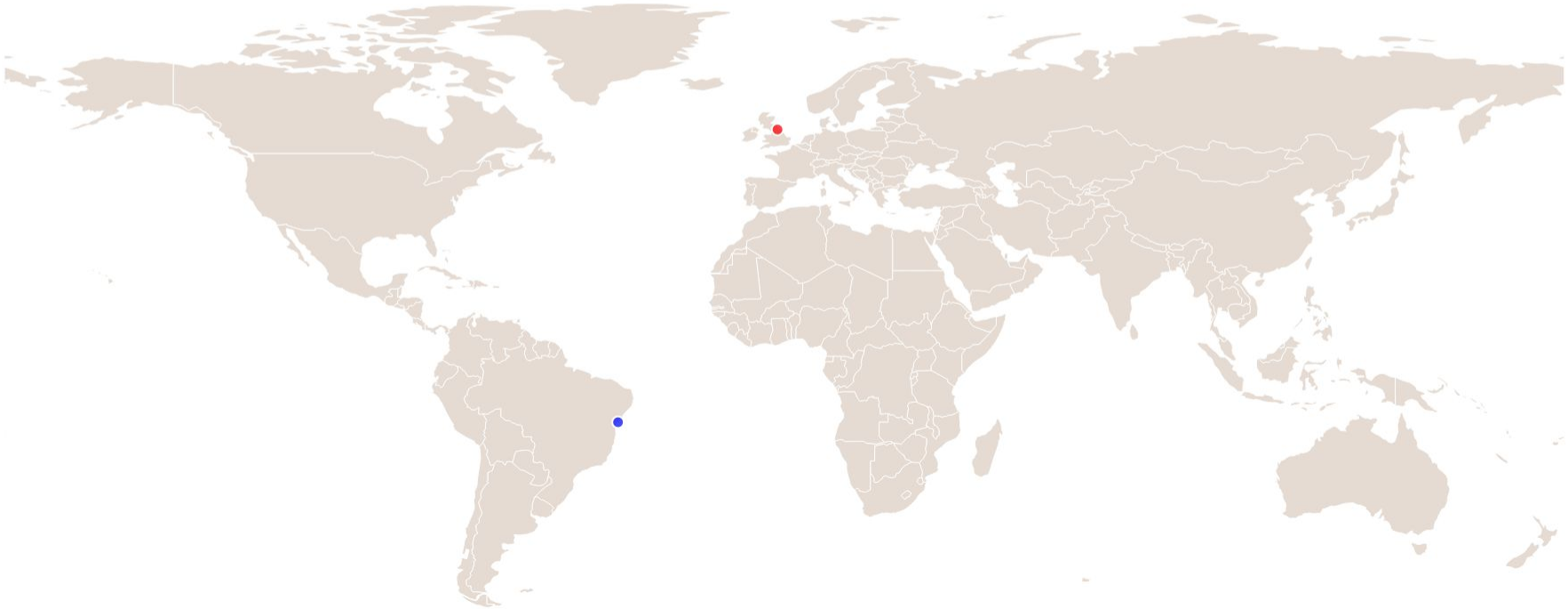
Flask Web Development | python - How to log ever | Response in python - Go | Response in python - You | Ude Dil Befikre - Song | S | Real Time Twitter Sentim | 0.0.0.0:9999/stream | DataMaps | D3.js tutorial - 2 - Select

0.0.0.0:9999

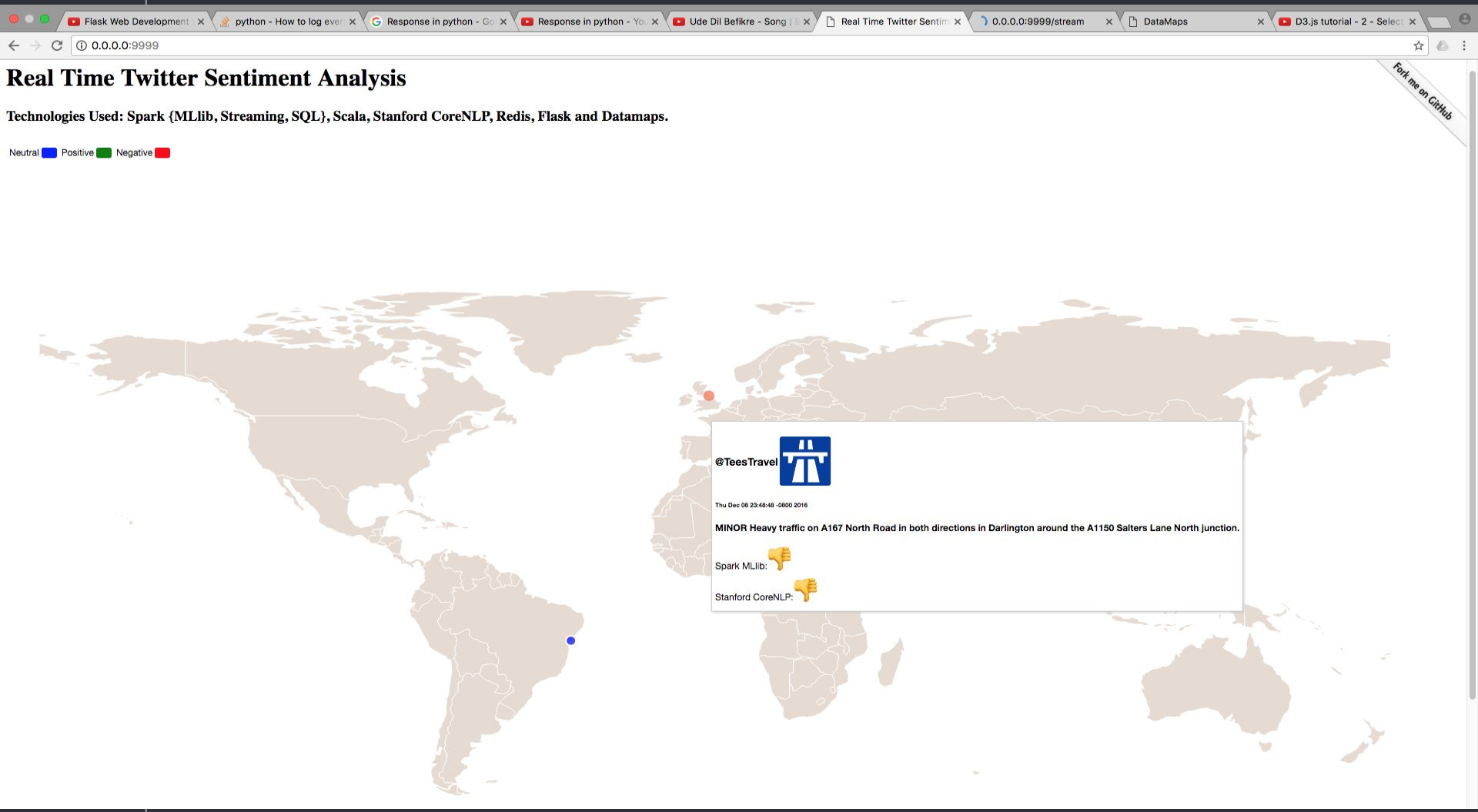
Real Time Twitter Sentiment Analysis

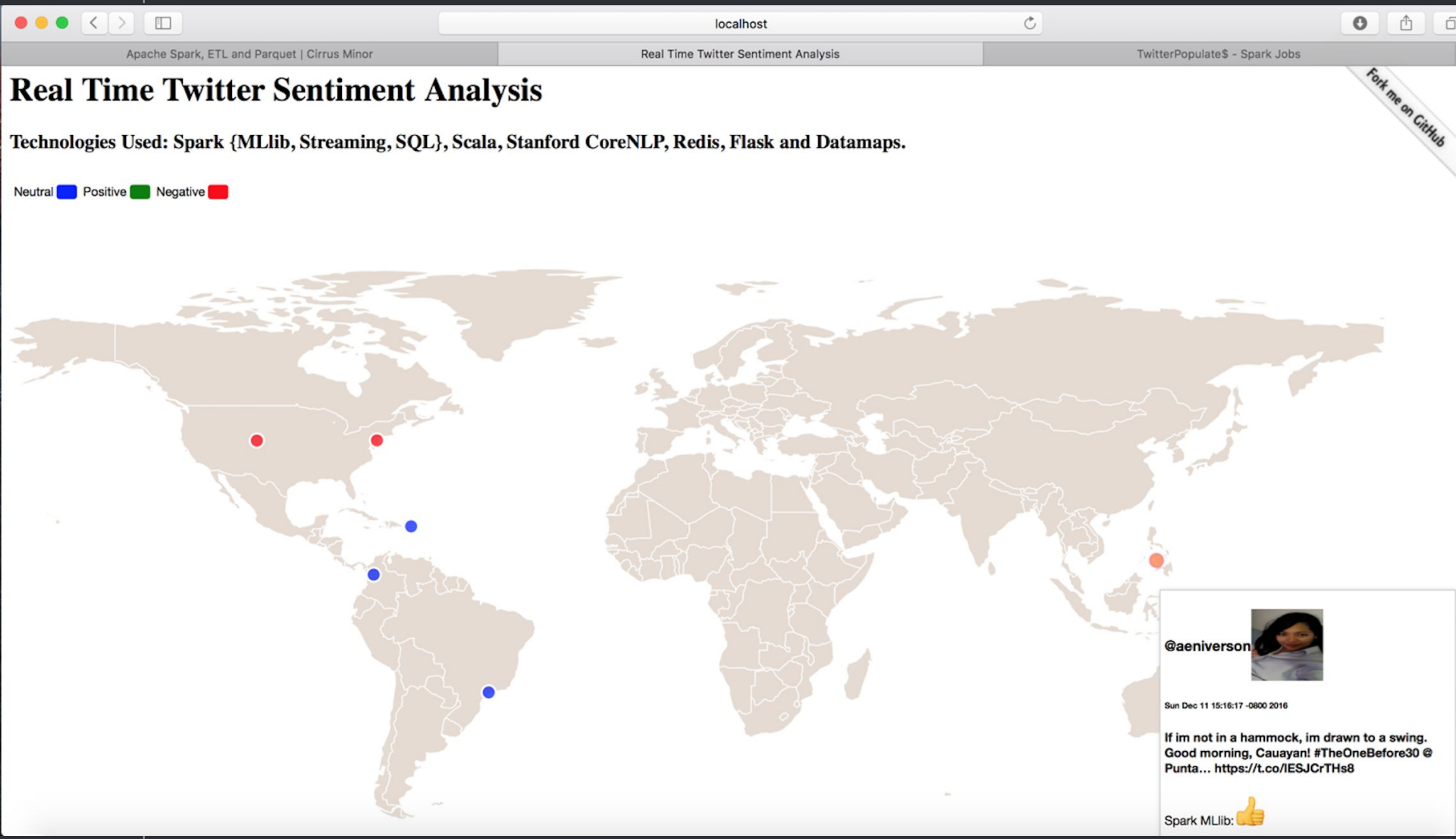
Technologies Used: Spark {MLlib, Streaming, SQL}, Scala, Stanford CoreNLP, Redis, Flask and Datamaps.

Neutral Positive Negative



Fork me on GitHub


Flask Web Development
python - How to log ever
Response in python - Go
Response in python - You
Ude Dil Befikre - Song |
Real Time Twitter Sentim
0.0.0.0:9999/stream
DataMaps
D3.js tutorial - 2 - Select
0.0.0.0:9999
Real Time Twitter Sentiment Analysis
Technologies Used: Spark {MLlib, Streaming, SQL}, Scala, Stanford CoreNLP, Redis, Flask and Datamaps.
Fork me on GitHub
Neutral
Positive
Negative
@TeesTravel
Thu Dec 08 23:48:48 -0800 2016
MINOR Heavy traffic on A167 North Road in both directions in Darlington around the A1150 Salters Lane North junction.
Spark MLlib:
Stanford CoreNLP:


localhost
Apache Spark, ETL and Parquet | Cirrus Minor
Real Time Twitter Sentiment Analysis
TwitterPopulate$ - Spark Jobs
Real Time Twitter Sentiment Analysis
Technologies Used: Spark {MLlib, Streaming, SQL}, Scala, Stanford CoreNLP, Redis, Flask and Datamaps.
Fork me on GitHub
Neutral
Positive
Negative
@aeniverson
Sun Dec 11 15:16:17 -0800 2016
If im not in a hammock, im drawn to a swing. Good morning, Cauayan! #TheOneBefore30 @ Punta… https://t.co/lESJCrTHs8
Spark MLlib:

Further work and Improvement Areas

- Update project using Apache spark 2.0
  - Utilize Data Frames and Datasets
  - Use machine learning package org.apache.spark.ml
- Find methods to process non english tweets
- Visualization can be improved

Thanks!

ANY QUESTIONS?