Exercise 8

Real-time Analysis of data using Spark Streaming

Prior Knowledge
Unix Command Line Shell
HDFS
Simple Python
Spark Python

Learning Objectives
Understand data delivery with Kafka
Parsing JSON with Spark
Understand micro-batch and Spark Streaming

Software Requirements
(see separate document for installation of these)

- Apache Spark 2.0.0
- Python 2.7.12
- Nano text editor or other text editor

I'll be honest, Spark Streaming is complex, and difficult to debug, so this lab guides you a lot of the way.

1. We have some data being fed to us via Kafak. You should be able to see this by using a Linux command line tool to subscribe to the feed:

    a. kafkacat –b kafka.freo.me –t tfl

    You should see lines like:
    ```
    {"stationId": "940GZZLUHBT", "trainNumber": "101",
    "stationName": "High Barnet Underground Station", "ttl":
    "2015-11-19T13:24:55.748Z", "timestamp": "2015-11-
    19T13:23:02.748Z", "line": "Northern", "expArrival":
    "2015-11-19T13:24:55.748Z"}
    ```

    Scrolling past every quite fast

2. This is data being read from Transport for London's API and then republished as JSON to Kafka. It concerns trains travelling on several of London's tube lines.

© Paul Fremantle 2015. Licensed under the This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. See http://creativecommons.org/licenses/by-nc-sa/4.0/

3. We are going to read this into Spark Streaming and calculate some data based on it. Our aim is to spot when a train's expected time changes (for the worse!).

4. Create a directory and a new file: (e.g ~/stream/tfl.streampy)

5. Copy the contents of the Python file from this URL into your local file: http://freo.me/ox-clo-kafka

6. First let's get this running and see if it works.

7. We need to use the Kafka streaming package, so this has to be specified.

8. From your directory, in a new terminal window type:

```
~/spark/bin/spark-submit \
--packages org.apache.spark:s-streaming-kafka-0-8_2.10:2.0.0 \
tflstream.py
```

9. Unfortunately you will see so much log that you won't be able to see the actual results! Stop that using Ctrl-C and now re-submit with the following:

```
~/spark/bin/spark-submit \
--packages org.apache.spark:s-streaming-kafka-0-8_2.10:2.0.0 \
tflstream.py 2>/dev/null
```

   This will redirect the stderr logs to /dev/null allowing us to see the output.

10. You should see something like:

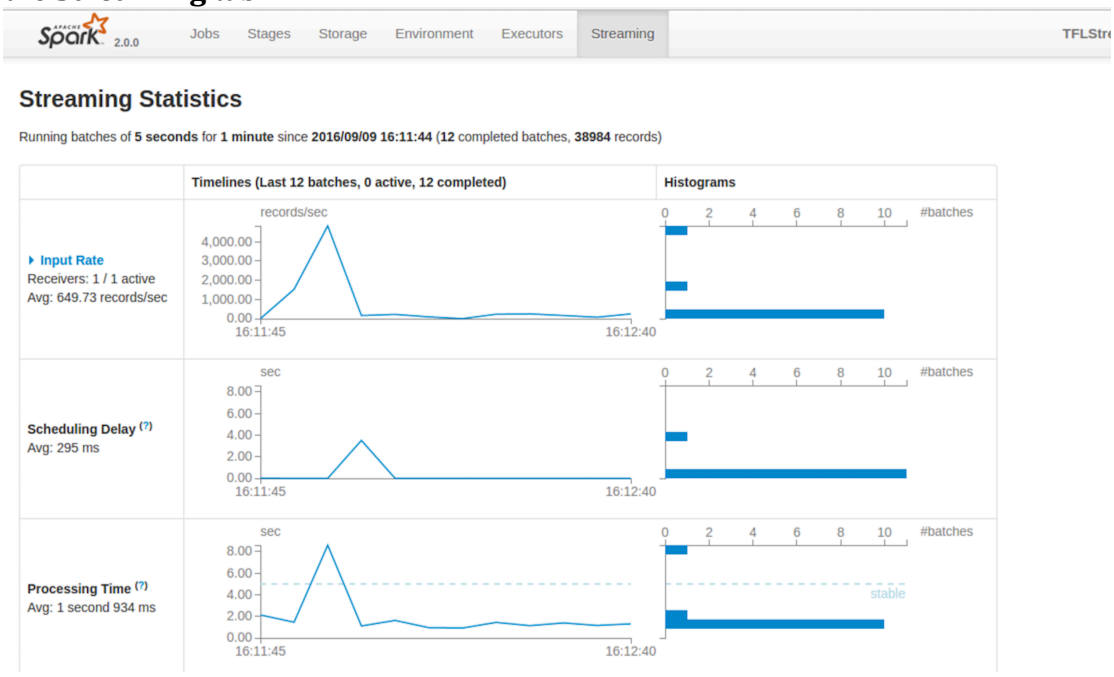
```
-------------------------------------------
Time: 2015-11-19 13:40:40
-------------------------------------------

-------------------------------------------
Time: 2015-11-19 13:40:45
-------------------------------------------

-------------------------------------------
Time: 2015-11-19 13:40:50
-------------------------------------------

-------------------------------------------
Time: 2015-11-19 13:40:55
-------------------------------------------
('delayed', 5)

-------------------------------------------
Time: 2015-11-19 13:41:00
-------------------------------------------
('delayed', 67)
('ontime', 2)

-------------------------------------------
Time: 2015-11-19 13:41:05
-------------------------------------------
('delayed', 68)
('ontime', 2)
```

© Paul Fremantle 2015. Licensed under the This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. See http://creativecommons.org/licenses/by-nc-sa/4.0/

11. I'm not sure this is completely fair to TfL. It seems to show most trains are delayed most of the time ☺ Either my code is dodgy or their predictions. What the code is doing is to save the previous estimate for each train to arrive at a given station and then compare the new estimate that comes in.

12. Take a look at the code. Work through each line and the comments.

13. Now take a look at the Spark console (http://localhost:4040), especially the **Streaming** tab:



14. Now let's see if you can modify the code:
    Can you change the code to print out how many ontime trains are due at each station.

15. Extension:
    Update the code to find the busiest station (with the most trains due)

© Paul Fremantle 2015. Licensed under the This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. See http://creativecommons.org/licenses/by-nc-sa/4.0/