W205 summer 2017 section 2 – Lab10 - K Iwasaki

**SUBMISSION 1:** Print only words with a length > 5 characters. Submit the pyspark code

MASTER=local[2] pyspark

from pyspark import SparkContext

from pyspark.streaming import StreamingContext

ssc = StreamingContext(sc, 1)

lines= ssc.textFileStream("file:///tmp/datastreams")

uclines = lines.map(lambda word: word.upper())

uclines = uclines.filter(lambda word: len(word) > 5)

uclines.pprint()

ssc.start()

ssc.stop()

**SUBMISSION 2:** Change the code so that you save the venue components to a text file. Submit you code.

MASTER=local[2] pyspark

from pyspark import SparkContext

from pyspark.streaming import StreamingContext

ssc = StreamingContext(sc, 1)

lines= ssc.textFileStream("file:///tmp/datastreams")

slines = lines.flatMap(lambda x: [ j['venue'] for j in json.loads('['+x+']') if 'venue' in j] )

cnt=slines.count()

cnt.pprint()

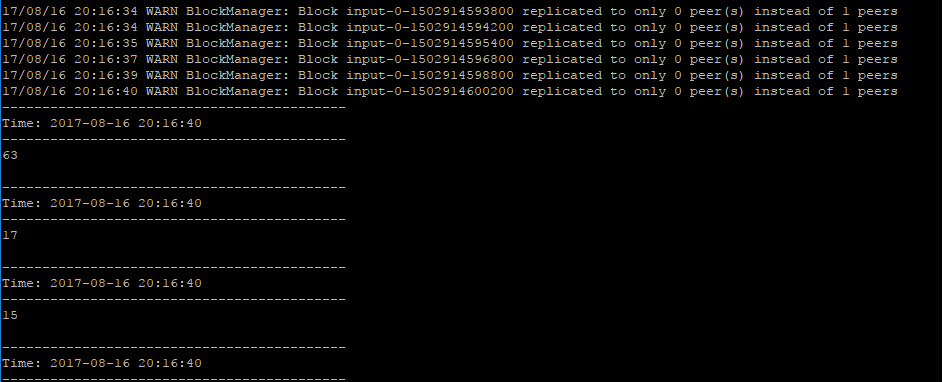
slines.pprint()

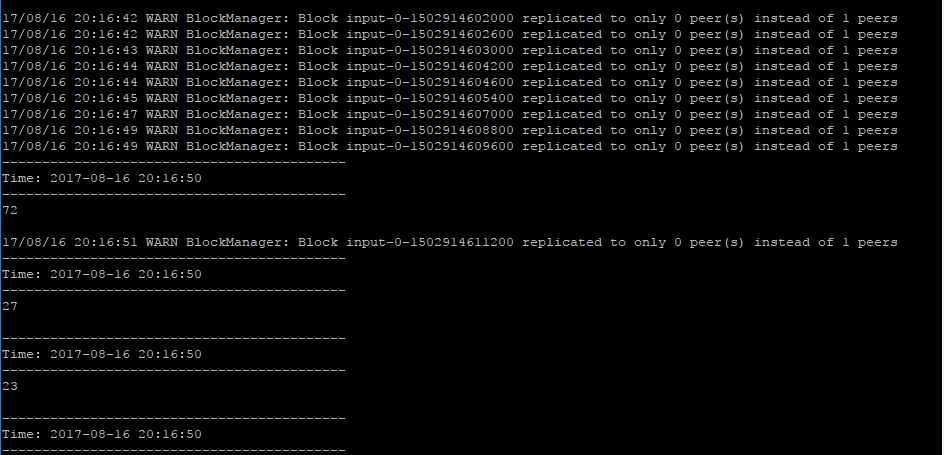
slines.saveAsTextFiles("file:///tmp/venues.txt")

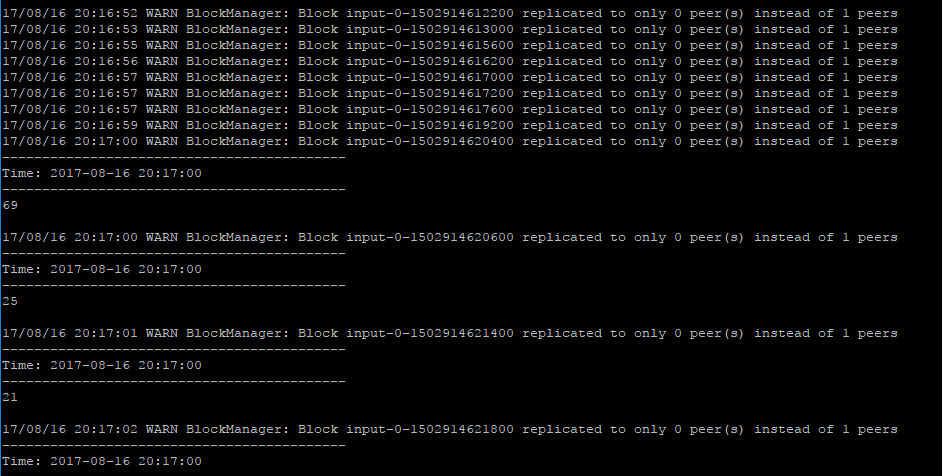
ssc.start()

ssc.stop()

**SUBMISSION 4a:** Provide a screenshot of the running Spark Streaming application that shows that the CountByWindow indeed provides an sum of the counts from the 3 latest batches.







**SUBMISSION 4b:** Also explain what the difference is between having 10 sec batches with a 30 sec sliding window and a 30 second batch length.

See the two columns: Batch: 30 sec and Window: 30 sec, Batch 10 sec for the difference

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Time**  **(base t)** | **Events**  Some random words stream | **Batch: 10 sec** | **Batch: 30 sec** | **Window: 30 sec,**  **Batch: 10 sec** |
| T | W1 | W1 |  |  |
| T+10 | W2 | W2 |  |  |
| T+20 | W3 | W3 | W1 W2 W3 | W1 W2 W3 |
| T+30 | W4 | W4 |  | W2 W3 W4 |
| T+40 | W5 | W5 |  | W3 W4 W5 |
| T+50 | W6 | W6 | W4 W5 W6 | W4 W5 W6 |
| T+60 | W7 | W7 |  | W5 W6 W7 |