ex62Bis

August 20, 2022

[ ]:
```python
from pyspark.streaming import StreamingContext
```

[ ]:
```python
# Create a Spark Streaming Context object
ssc = StreamingContext(sc, 30)
```

[ ]:
```python
# Create a (Receiver) DStream that will connect to localhost:9999
linesDStream = ssc.socketTextStream("localhost", 9999)
```

[ ]:
```python
# Computer for each stockID the price variation (compute it for each batch).
# Select only the stocks with a price variation (%) greater than 0.5%
```

[ ]:
```python
# Return one pair (stockId, (price, price) )  for each input record

def extractStockIdPricePrice(line):
    fields = line.split(",")

    stockId = fields[1]
    price = fields[2]

    return (stockId, (float(price), float(price)) )


stockIdPriceDStream = linesDStream.map(extractStockIdPricePrice)
```

[ ]:
```python
# Compute max and min for each stockId
# Set the windows zise to 60 seconds
# The sliding interval is equal to 30 seconds, i.e., 1 batch
stockIdMaxMinPriceDStream = stockIdPriceDStream\
.reduceByKeyAndWindow(lambda v1, v2: ( max(v1[0],v2[0]), min(v1[1],v2[1]) ),␣
 ↪None, 60)
```

[ ]:
```python
# Compute variation for each stock
stockIdVariationDStream = stockIdMaxMinPriceDStream\
.mapValues(lambda MaxMinValue: 100.0*(MaxMinValue[0]-MaxMinValue[1])/
 ↪MaxMinValue[0] )
```

[ ]:

```
# Select only the stocks with variation greater than 0.5%
selectedStockIdsVariationsDStream = stockIdVariationDStream.filter(lambda pair: pair[1]>0.5)
```

[ ]:

```
selectedStockIdsVariationsDStream.pprint()
```

[ ]:

```
#Start the computation
ssc.start()
```

[ ]:

```
# Run this application for 200 seconds
ssc.awaitTerminationOrTimeout(200)
ssc.stop(stopSparkContext=False)
```

[ ]:

```

```