TP Spark : Spark batch and stream processing: Twitter Sentiment Analysis

Mastère Big Data 2019/2020

Benjamin Thery - benjamin.thery@grenoble-inp.org

1. Batch Processing

Fichier Source : BatchAnalysis.scala

1.1 Combien de tweets mentionnent Donald Trump ?

```
val tweets: RDD[String] = sc.textFile("1Mtweets_en.txt")
val donaldTweets: RDD[String] = tweets.filter(text => text.contains("Trump"))

println("==== 'Trump' mentions in tweets ====")
println(donaldTweets.count())
```

1.2 Construire un RDD contenant des paires (tweet, sentiment) indiquant le sentiment de chaque tweet.

```
val tweets: RDD[String] = sc.textFile("1Mtweets_en.txt")
val tweetSentiments: RDD[(String, Double)] = tweets.map(tweet => (tweet,
TweetUtilities.getSentiment(tweet)))

println("==== Sentiments of 5 tweets ====")
tweetSentiments.take(5).foreach(println)
```

1.3 En utilisant le RDD précédent, créer un RDD représentant le sentiment associé à chaque hashtag (ou mention)

Pour représenter la sentiment associé à chaque tag, j'ai pris la moyenne des sentiments de tous les tweets utilisant le tag.

```
// 2.3. Sentiments associated with each hashtags: average of tweet sentiment values
associated with each hashtag

// For each tweet, return a list of (hashtag, sentiment value)
def tweetToHashtagSentiment(tweet: String ) : List[(String, Double)] = {
    val sentiment: Double = TweetUtilities.getSentiment(tweet)

    val lb: ListBuffer[(String, Double)] = ListBuffer()

    for (ht <- TweetUtilities.getHashTags(tweet)) {
        lb += ((ht, sentiment))
    }
    return lb.toList
}

def myAverage(buf: Iterable[Double]) : Double = {
```

```
    var mysum = 0.0
    buf.foreach(mysum += _)
    return mysum / buf.size
}

// RDD extraction of all tags from their tweet,
// the value is the sentiment of the original tweet .
// There are duplicate keys in the RDD (hashtag)
val hashtagSentiment: RDD[(String, Double)] = tweets.flatMap(tw =>
tweetToHashtagSentiment(tw))

// RDD that groups the entry of the previous RDD by key (hashtag)
// Each hashtag has a list of sentiments values
val hashtagGrouped = hashtagSentiment.groupByKey()

// RDD for each hashtag compute the average sentiment associated
// with it by mapping its values
val hashtagSentimentFinal = hashtagGrouped.map(x => (x._1, myAverage(x._2)))


println("==== Hashtags Sentiments for 10 tags ====")
hashtagSentimentFinal.take(10).foreach(println)
```

1.4 Afficher les K plus positifs / négatifs hashtags

En partant du dernier RDD de la question précédente :

```
// 2.4 most positive/negative hashtags

println("==== Most positive hashtags ====")
hashtagSentimentFinal.sortBy(_._2, ascending=false).take(10).foreach(println)

println("==== Most negative hashtags ====")
hashtagSentimentFinal.sortBy(_._2, ascending=true).take(10).foreach(println)
```

2. Stream Processing