Spark Cheatsheet

Contents

[Spark Cheatsheet 1](#_Toc442593205)

[Pair RDDs 1](#_Toc442593206)

[Count the occurrences of a word 1](#_Toc442593207)

[Count the number of unique words 2](#_Toc442593208)

[Extract the Keys and Values from an RDD 2](#_Toc442593209)

[Return as a list 2](#_Toc442593210)

[Find the n most common words 2](#_Toc442593211)

[groupByKey() 3](#_Toc442593212)

[Keep only unique values in a Key-Value PairRDD 3](#_Toc442593213)

[Sorting by Keys and Values 3](#_Toc442593214)

[Resilient Distributed Datasets (RDDs) 4](#_Toc442593215)

[Extract the first n data points from an RDD 4](#_Toc442593216)

[Number of elements in the RDD 4](#_Toc442593217)

[Read a file into an RDD 4](#_Toc442593218)

# Pair RDDs

## Count the occurrences of a word

In:

wordsList = ['cat', 'elephant', 'rat', 'rat', 'cat']

wordsRDD = sc.parallelize(wordsList, 4)

Out:

['cat', 'elephant', 'rat', 'rat', 'cat']

In:

wordPairs = wordsRDD.map(lambda x : (x,1))

Out:

[('cat', 1), ('elephant', 1), ('rat', 1), ('rat', 1), ('cat', 1)]

In:

wordCounts = wordPairs.reduceByKey(lambda a,b : a+b)

Out:

[('rat', 2), ('elephant', 1), ('cat', 2)]

## Count the number of unique words

In:

uniqueWords = len(wordCounts.collect())

print uniqueWords

Out:

3

## Extract the Keys and Values from an RDD

keys()

values()

### Return as a list

daysWithHosts = dailyHosts.keys().collect()

## Find the n most common words

endpointCountPairTuple = not200.map(lambda log: (log.endpoint, 1))

endpointSum = endpointCountPairTuple.reduceByKey(lambda a,b : a + b)

# these endpointSum tuples are like

# [(u'/images/NASA-logosmall.gif', 8761), (u'/images/KSC-logosmall.gif', 7236),…]

print endpointSum.takeOrdered(10, lambda s: -1 \* s[1])

## groupByKey()

This outputs a ResultIterable object , which can be cast to a list using a .mapValues() function

dayAndHostTuple = access\_logs.map(lambda log: (log.date\_time.day, log.host))

print dayAndHostTuple.take(3)

groupedByDay = dayAndHostTuple.groupByKey()

print groupedByDay.take(3)

output

[(1, u'in24.inetnebr.com'), (1, u'uplherc.upl.com'), (1, u'uplherc.upl.com')]

[(8, <pyspark.resultiterable.ResultIterable object at 0xb0904bec>), (12, <pyspark.resultiterable.ResultIterable object at 0xb0904a6c>), (4, <pyspark.resultiterable.ResultIterable object at 0xb0902a2c>)]

If you want the value of the PairRDD to be a list, use:

groupedByDay = (dayAndHostTuple.groupByKey()

.mapValues(lambda x: list(x)))

output:

[(8, [u'pc7567.dialup.rwth-aachen.de', u'mtrsppp46.epix.net',…]), (9, […]), …]

## Keep only unique values in a Key-Value PairRDD

Using list(set(x)) converts the iterable result of groupByKey() to a list containing only the distinct values.

dayGroupedHosts = (dayToHostPairTuple.groupByKey()

.mapValues(lambda x: list(set(x))))

## Sorting by Keys and Values

dayHostCount = dayGroupedHosts.mapValues(lambda host\_list: len(host\_list))

# This looks like [(8, 60142), (12, 38070), (4, 59554), (16, 56651), (20, 32963)]

**dailyHosts = dayHostCount.sortByKey()**

print dailyHosts.take(3)

output: [(1, 33996), (3, 41387), (4, 59554)]

Sort by keys (ascending):

RDD.takeOrdered(5, key = lambda x: x[0])

Sort by keys (descending):

RDD.takeOrdered(5, key = lambda x: -x[0])

Sort by values (ascending):

RDD.takeOrdered(5, key = lambda x: x[1])

Sort by values (descending):

RDD.takeOrdered(5, key = lambda x: -x[1])

# Resilient Distributed Datasets (RDDs)

## Extract the first n data points from an RDD

samplePoints = rawData.take(n)

Filter an RDD

## Number of elements in the RDD

numPoints = rawData.count()

print numPoints

## Read a file into an RDD

from test\_helper import Test

import os.path

baseDir = os.path.join('data')

inputPath = os.path.join('cs190', 'millionsong.txt')

fileName = os.path.join(baseDir, inputPath)

numPartitions = 2

rawData = sc.textFile(fileName, numPartitions)