Setup Notebook for Exercises

IMPORTANT: Only modify cells which have the following comment:

```
# Modify this cell
```

Do not add any new cells when you submit the homework

In [1]:
```
import findspark
findspark.init()
```

In [2]:
```
from pyspark import SparkContext
sc=SparkContext(master="local[4]")
```

In [3]:
```
import Tester.WordCount as WordCount
pickleFile="Tester/WordCount.pkl"
```

Importing all packages necessary to complete the homework

In [4]:
```
import numpy as np
```

In [5]:
```
WordCount.get_data()
```

Exercise

A `k`-mer is a sequence of `k` consecutive words.

For example, the `3`-mers in the line `you are my sunshine my only sunsine` are

- `you are my`
- `are my sunshine`
- `my sunshine my`
- `sunshine my only`
- `my only sunsine`

For the sake of simplicity we consider only the `k`-mers that appear in a single line. In other words, we ignore `k`-mers that span more than one line.

Write a function, using spark all the way to the end, to find to top 10 `k`-mers in a given text for a given `k`.

Specifically write functions with the following signatures:

```
    def map_kmers(text,k):
        \\ text: an RDD of text lines. Lines contain only lower-case let
    ters and spaces. Spaces should be ignored.
        \\ k: length of `k`-mers
        return  singles
        \\ singles: an RDD of pairs of the form (tuple of k words,1)
    def count_kmers(singles):
        \\ singles: as above
        return counts
        \\ count: RDD of the form: (tuple of k words, number of occuranc
    es)
    def sort_counts(count):
        \\ count: as above
        return sorted_counts
        \\ sorted_counts: RDD of the form (number of occurances, tuple o
    f k words) sorted in decreasing number of occurances.
```

Code:

```
    text_file = sc.textFile(u'../../Data/Moby-Dick.txt')
    print getkmers(text_file,5,2, map_kmers, count_kmers, sort_counts)
```

Output:

most common 2-mers
1616: (u'of', u'the')
1019: (u'in', u'the')
635: (u'to', u'the')
381: (u'from', u'the')
335: (u'the', u'whale')

In [169]:
```
def map_kmers(text,k):
    # text: an RDD of text lines. Lines contain only lower-case letters and
    # k: length of `k`-mers
    lines =text.map(lambda line: line.split())
    lines1=lines.filter(lambda x: x!='')
    singles = lines1.flatMap(lambda x: [(x[i:],1) for i in range(k)])
    return  singles
    # singles: an RDD of pairs of the form (tuple of k words,1)
def count_kmers(singles):
    # singles: as above
    count = singles.reduceByKey(lambda x,y:x+y)
    return count
    # count: RDD of the form: (tuple of k words, number of occurances)
def sort_counts(count):
    # count: as above
    sorted_count = count.map(lambda x:(x[1],x[0])).sortByKey(False)
    return sorted_count
    # sorted_counts: RDD of the form (number of occurances, tuple of k words
```

In [170]:
```
# Do Not modify this cell
def getkmers(text_file, l,k, map_kmers, count_kmers, sort_counts):
    # text_file: the text_file RDD read above
    # k: k-mers
    # l: l most common k-mers

    import re
    def removePunctuation(text):
        return re.sub("[^0-9a-zA-Z ]", " ", text)
    text = text_file.map(removePunctuation)\
                    .map(lambda x: x.lower())

    singles=map_kmers(text,k)
    count=count_kmers(singles)
    sorted_counts=sort_counts(count)

    C=sorted_counts.take(l)
    print 'most common %d-mers\n'%k,'\n'.join(['%d:\t%s'%c for c in C])
```

In [171]:
```
# First, check that the text file is where we expect it to be
%ls -l ../../Data/Moby-Dick.txt
```

```
-rw-r--r--  1 xiasong  staff  1257260 Apr 21 11:21 ../../Data/Moby-Dick.t
xt
```

In [172]:
```
text_file = sc.textFile(u'../../Data/Moby-Dick.txt')
```

In [173]:
```
# Print the output of the aggregate function for top 5 2-mers
getkmers(text_file,5,2, map_kmers, count_kmers, sort_counts)
```

```
---------------------------------------------------------------------------
--
Py4JJavaError                             Traceback (most recent call las
t)
<ipython-input-173-3e66ddaf0859> in <module>()
      1 # Print the output of the aggregate function for top 5 2-mers
----> 2 getkmers(text_file,5,2, map_kmers, count_kmers, sort_counts)

<ipython-input-170-10bd8d9f2b0f> in getkmers(text_file, l, k, map_kmers,
 count_kmers, sort_counts)
     12     singles=map_kmers(text,k)
     13     count=count_kmers(singles)
---> 14     sorted_counts=sort_counts(count)
     15
     16     C=sorted_counts.take(l)

<ipython-input-169-b811ef527203> in sort_counts(count)
     17 def sort_counts(count):
     18     # count: as above
   > 19     sorted_count = count.map(lambda x:(x[1],x[0])).sortByKey(Fals
```

In [91]:
```
import Tester.WordCount as WordCount
WordCount.exercise(pickleFile, map_kmers, count_kmers, sort_counts, sc)
```

```
PythonRDD[181] at RDD at PythonRDD.scala:48
Correct Output: [(78, (u'the', u'sperm', u'whale')), (73, (u'of', u'the',
u'whale')), (65, (u'the', u'white', u'whale')), (51, (u'one', u'of', u'th
e')), (48, (u'of', u'the', u'sea'))]

Error: Function returned incorrect output
Your Output:  [(1796, (u'of', u'the')), (1145, (u'in', u'the')), (708,
 (u'to', u'the')), (408, (u'from', u'the')), (376, (u'the', u'whale'))]

---------------------------------------------------------------------------
--
AssertionError                            Traceback (most recent call las
t)
<ipython-input-91-2f960e215751> in <module>()
      1 import Tester.WordCount as WordCount
----> 2 WordCount.exercise(pickleFile, map_kmers, count_kmers, sort_count
s, sc)

/Users/xiasong/Documents/Class_2016/DSE/CSE255-DSE230/Classes/1.SparkBasi
c/Tester/WordCount.pyc in exercise(pickleFile, map_kmers, count_kmers, so
rt_counts, sc)
     30     func_student = lambda RDD: getkmers(RDD, 5,3, map_kmers, coun
t_kmers, sort_counts)
     31     noError = TestRDDK( data=text_file,
func_student=func_student, corAns=case[0], corType=case[1], takeK=5, toPr
int=False)
---> 32     if noError == False: raise AssertionError('Your Answer is Inc
orrect')
     33     print

AssertionError: Your Answer is Incorrect
```

In [ ]:
```

```