Spark Developer Training - 3 Days

Manaranjan Pradhan
manaranjan@enablecloud.com

Analyze the captains performance in ODIs and Test matches in Cricket

This notebook gives on overview of how to read data into Spark Framework and apply some basic RDD operations

In [1]:

```
# Spark Context Created.. Let's verify it
sc
```

Out[1]:

```
<pyspark.context.SparkContext at 0x7fa5801ad278>
```

ODI Performance Analysis

Read the capatings_ODI.csv file from local file system

In [2]:

```
captains_odis = sc.textFile( "file:///home/hadoop/lab/data/captains_ODI.csv" )
```

Check the first record

In [3]:

```
captains_odis.first()
```

Out[3]:

```
'Ponting  R T,Australia,1995-2012,230,165,51,14,124'
```

Display the first 10 records

In [4]:

```
captains_odis.take( 10 )
```

Out[4]:

```
['Ponting  R T,Australia,1995-2012,230,165,51,14,124',
 'Fleming  S P,New Zealand,1994-2007,218,98,106,14,105',
 'Ranatunga  A,Sri Lanka,1982-1999,193,89,95,9,102',
 'Dhoni  M S*,India,2004-,186,103,68,15,88',
 'Border  A R,Australia,1979-1994,178,107,67,4,86',
 'Azharuddin  M,India,1985-2000,174,89,77,8,96',
 'Smith  G C,South Africa,2002-2013,149,91,51,7,74',
 'Ganguly  S C,India,1992-2007,147,76,66,5,74',
 'Cronje  W J,South Africa,1992-2000,140,99,37,4,74',
 'Imran Khan,Pakistan,1974-1992,139,75,59,5,70']
```

Checking the RDD type

In [5]:

```
# captains_odis should be of RDD type
type( captains_odis )
```

Out[5]:

```
pyspark.rdd.RDD
```

Counting the number of records

In [6]:

```
# Count the number of records
captains_odis.count()
```

Out[6]:

```
98
```

Parsing the lines and creating records

The RDD does not have a header. Let's define a header and then assign it to the RDD

In [7]:

```
fields = ("name", "country", "career", "matches", "won", "lost", "ties", "toss" )
```

In [8]:

```
from collections import namedtuple
```

Lets define a tuple (a record) for each line. We need to iterate through each line and convert that into a record. The record can be defined as a namedTuple type and called Captain. Let's also link the names for the fields

In [9]:

```
Captain = namedtuple( 'Captain', fields )
```

Create a function to parse the lines and create namedtuples

In [10]:

```
# Define a function to parse each line and convert them into records
def parseRecs( line ):
    fields = line.split(",")
    return Captain( fields[0], fields[1], fields[2], int( fields[3] ),
                   int( fields[4] ), int(fields[5]), int(fields[6]), int(fields[7] )
```

Iterate throught the data and convert them into records

In [11]:

```
captains = captains_odis.map( lambda rec: parseRecs( rec) )
```

In [12]:

```
# Now captains refer to all the records. Let's display the first 10 records.
captains.take( 10 )
```

Out[12]:

```
[Captain(name='Ponting  R T', country='Australia', career='1995-2012',
matches=230, won=165, lost=51, ties=14, toss=124),
 Captain(name='Fleming  S P', country='New Zealand', career='1994-200
7', matches=218, won=98, lost=106, ties=14, toss=105),
 Captain(name='Ranatunga  A', country='Sri Lanka', career='1982-1999',
matches=193, won=89, lost=95, ties=9, toss=102),
 Captain(name='Dhoni  M S*', country='India', career='2004-', matches=1
86, won=103, lost=68, ties=15, toss=88),
 Captain(name='Border  A R', country='Australia', career='1979-1994', m
atches=178, won=107, lost=67, ties=4, toss=86),
 Captain(name='Azharuddin  M', country='India', career='1985-2000', mat
ches=174, won=89, lost=77, ties=8, toss=96),
 Captain(name='Smith  G C', country='South Africa', career='2002-2013',
matches=149, won=91, lost=51, ties=7, toss=74),
 Captain(name='Ganguly  S C', country='India', career='1992-2007', matc
hes=147, won=76, lost=66, ties=5, toss=74),
 Captain(name='Cronje  W J', country='South Africa', career='1992-200
0', matches=140, won=99, lost=37, ties=4, toss=74),
 Captain(name='Imran Khan', country='Pakistan', career='1974-1992', mat
ches=139, won=75, lost=59, ties=5, toss=70)]
```

In [13]:

```
# What is the type of the captains RDD
type( captains )
```

Out[13]:

```
pyspark.rdd.PipelinedRDD
```

Filtering records

Filter only those captains that have captained for at least 100 ODI matches. And then we can compare the statistics of these captains

In [14]:

```
captains_100 = captains.filter( lambda rec: rec.matches > 100 )
```

How many captains have captained their country for more than 100 ODIs?

In [15]:

```
captains_100.count()
```

Out[15]:

```
16
```

In [16]:

```
# Who are these captains
captains.take( 10 )
```

Out[16]:

```
[Captain(name='Ponting  R T', country='Australia', career='1995-2012',
matches=230, won=165, lost=51, ties=14, toss=124),
 Captain(name='Fleming  S P', country='New Zealand', career='1994-200
7', matches=218, won=98, lost=106, ties=14, toss=105),
 Captain(name='Ranatunga  A', country='Sri Lanka', career='1982-1999',
matches=193, won=89, lost=95, ties=9, toss=102),
 Captain(name='Dhoni  M S*', country='India', career='2004-', matches=1
86, won=103, lost=68, ties=15, toss=88),
 Captain(name='Border  A R', country='Australia', career='1979-1994', m
atches=178, won=107, lost=67, ties=4, toss=86),
 Captain(name='Azharuddin  M', country='India', career='1985-2000', mat
ches=174, won=89, lost=77, ties=8, toss=96),
 Captain(name='Smith  G C', country='South Africa', career='2002-2013',
matches=149, won=91, lost=51, ties=7, toss=74),
 Captain(name='Ganguly  S C', country='India', career='1992-2007', matc
hes=147, won=76, lost=66, ties=5, toss=74),
 Captain(name='Cronje  W J', country='South Africa', career='1992-200
0', matches=140, won=99, lost=37, ties=4, toss=74),
 Captain(name='Imran Khan', country='Pakistan', career='1974-1992', mat
ches=139, won=75, lost=59, ties=5, toss=70)]
```

In [17]:

```
# Filtering: Captains who have more wins then losses...
captains_more_wins = captains_100.filter( lambda rec: rec.won > rec.lost )
```

Collecting the results in the driver

In [18]:

```
# Captains with more wins than losses
captains_more_wins.map( lambda rec: rec.name ).collect()
```

Out[18]:

```
['Ponting  R T',
 'Dhoni  M S*',
 'Border  A R',
 'Azharuddin  M',
 'Smith  G C',
 'Ganguly  S C',
 'Cronje  W J',
 'Imran Khan',
 'Jayawardene  D P M',
 'Jayasuriya  S T',
 'Wasim Akram',
 'Waugh  S R',
 'Richards  I V A']
```

In [19]:

```
# Captains with less wins than losses
captains_more_losts = captains_100.filter( lambda rec: rec.won <= rec.lost )
captains_more_losts.map( lambda rec: rec.name ).collect()
```

Out[19]:

```
['Fleming  S P', 'Ranatunga  A', 'Lara  B C']
```

Creating a subset of data by filtering columns

In [20]:

```
# Which country has played how many matches..
countries = captains.map( lambda rec: ( rec.country , rec.matches) )
```

In [21]:

```
countries.take( 10 )
```

Out[21]:

```
[('Australia', 230),
 ('New Zealand', 218),
 ('Sri Lanka', 193),
 ('India', 186),
 ('Australia', 178),
 ('India', 174),
 ('South Africa', 149),
 ('India', 147),
 ('South Africa', 140),
 ('Pakistan', 139)]
```

Aggregating values by keys

In [22]:

```
# Aggregate by countries
matches_countries = countries.reduceByKey( lambda a, b: a + b )
```

In [23]:

```
matches_countries.take( 20 )
```

Out[23]:

```
[('Pakistan', 781),
 ('South Africa', 463),
 ('Ireland', 93),
 ('Australia', 832),
 ('West Indies', 658),
 ('Kenya', 114),
 ('India', 770),
 ('Bermuda', 31),
 ('Netherlands', 31),
 ('Afghanistan', 50),
 ('England', 554),
 ('Canada', 27),
 ('Zimbabwe', 394),
 ('Sri Lanka', 710),
 ('Bangladesh', 251),
 ('New Zealand', 608)]
```

In [24]:

```
# Sort the countries by the number of matches they played.
# Sort by names...(sort by key)
matches_countries.sortByKey().collect()
```

Out[24]:

```
[('Afghanistan', 50),
 ('Australia', 832),
 ('Bangladesh', 251),
 ('Bermuda', 31),
 ('Canada', 27),
 ('England', 554),
 ('India', 770),
 ('Ireland', 93),
 ('Kenya', 114),
 ('Netherlands', 31),
 ('New Zealand', 608),
 ('Pakistan', 781),
 ('South Africa', 463),
 ('Sri Lanka', 710),
 ('West Indies', 658),
 ('Zimbabwe', 394)]
```

Sorting records

In [25]:

```
# Sort the countries by the number of matches they played by descending order..
# by names (key)
matches_countries.sortByKey( ascending = False ).collect()
```

Out[25]:

```
[('Zimbabwe', 394),
 ('West Indies', 658),
 ('Sri Lanka', 710),
 ('South Africa', 463),
 ('Pakistan', 781),
 ('New Zealand', 608),
 ('Netherlands', 31),
 ('Kenya', 114),
 ('Ireland', 93),
 ('India', 770),
 ('England', 554),
 ('Canada', 27),
 ('Bermuda', 31),
 ('Bangladesh', 251),
 ('Australia', 832),
 ('Afghanistan', 50)]
```

In [26]:

```
# Sort by values.. default is ascending...
matches_countries.sortBy( lambda rec: rec[1] ).collect()
```

Out[26]:

```
[('Canada', 27),
 ('Bermuda', 31),
 ('Netherlands', 31),
 ('Afghanistan', 50),
 ('Ireland', 93),
 ('Kenya', 114),
 ('Bangladesh', 251),
 ('Zimbabwe', 394),
 ('South Africa', 463),
 ('England', 554),
 ('New Zealand', 608),
 ('West Indies', 658),
 ('Sri Lanka', 710),
 ('India', 770),
 ('Pakistan', 781),
 ('Australia', 832)]
```

In [27]:

```
# Sort by value by descending
matches_countries.sortBy( lambda rec: rec[1], ascending = False ).collect()
```

Out[27]:

```
[('Australia', 832),
 ('Pakistan', 781),
 ('India', 770),
 ('Sri Lanka', 710),
 ('West Indies', 658),
 ('New Zealand', 608),
 ('England', 554),
 ('South Africa', 463),
 ('Zimbabwe', 394),
 ('Bangladesh', 251),
 ('Kenya', 114),
 ('Ireland', 93),
 ('Afghanistan', 50),
 ('Bermuda', 31),
 ('Netherlands', 31),
 ('Canada', 27)]
```

In [28]:

```
# Captains by percentage of wins
captains_100_percent_wins = captains_100.map(
    lambda rec: ( rec.name, round( rec.won/rec.matches, 2 ) ) )

# Sort by percentage wins
captains_100_percent_wins.sortBy(
    lambda rec: rec[1], ascending = False ).collect()
```

Out[28]:

```
[('Ponting  R T', 0.72),
 ('Cronje  W J', 0.71),
 ('Richards  I V A', 0.64),
 ('Waugh  S R', 0.63),
 ('Smith  G C', 0.61),
 ('Wasim Akram', 0.61),
 ('Border  A R', 0.6),
 ('Jayasuriya  S T', 0.56),
 ('Dhoni  M S*', 0.55),
 ('Jayawardene  D P M', 0.55),
 ('Imran Khan', 0.54),
 ('Ganguly  S C', 0.52),
 ('Azharuddin  M', 0.51),
 ('Lara  B C', 0.47),
 ('Ranatunga  A', 0.46),
 ('Fleming  S P', 0.45)]
```

Exercise for the participants

- Filter countries which have played more than hundred matches
- Sort counties by the percentage of matches they
- Find the top10 lucky captains in terms of TOSS w

In [29]:

```
# Read the Captains_Test.csv file
captains_tests = sc.textFile( "file:///home/hadoop/lab/data/captains_Test.csv" )
```

In [30]:

```
# Parse the records
captains_tests_recs = captains_tests.map( lambda rec: parseRecs( rec ) )
```

In [31]:

```
# Display the first 10 records
captains_tests_recs.take( 10 )
```

Out[31]:

```
[Captain(name='Smith  G C', country='South Africa', career='2002-2014',
matches=109, won=53, lost=29, ties=27, toss=58),
 Captain(name='Border  A R', country='Australia', career='1978-1994', m
atches=93, won=32, lost=22, ties=38, toss=47),
 Captain(name='Fleming  S P', country='New Zealand', career='1994-200
8', matches=80, won=28, lost=27, ties=25, toss=38),
 Captain(name='Ponting  R T', country='Australia', career='1995-2012',
matches=77, won=48, lost=16, ties=13, toss=37),
 Captain(name='Lloyd  C H', country='West Indies', career='1966-1984',
matches=74, won=36, lost=12, ties=26, toss=35),
 Captain(name='Dhoni  M S*', country='India', career='2005-', matches=6
0, won=27, lost=18, ties=15, toss=27),
 Captain(name='Waugh  S R', country='Australia', career='1985-2004', ma
tches=57, won=41, lost=9, ties=7, toss=31),
 Captain(name='Ranatunga  A', country='Sri Lanka', career='1982-2000',
matches=56, won=12, lost=19, ties=25, toss=29),
 Captain(name='Atherton  M A', country='England', career='1989-2001', m
atches=54, won=13, lost=21, ties=20, toss=23),
 Captain(name='Cronje  W J', country='South Africa', career='1992-200
0', matches=53, won=27, lost=11, ties=15, toss=22)]
```

In [32]:

```
# Filter the captains who have captained for more than 100 tests
captains_tests_100 = captains_tests_recs.filter( lambda rec:
                                                rec.matches > 100 )
```

In [33]:

```
## How many captains?
captains_tests_100.take( 10 )
```

Out[33]:

```
[Captain(name='Smith  G C', country='South Africa', career='2002-2014',
matches=109, won=53, lost=29, ties=27, toss=58)]
```

In [34]:

```
# Filter the captains who have captained for more than 50 tests
captains_tests_50 = captains_tests_recs.filter( lambda rec:
                                               rec.matches > 50 )
```

In [35]:

```
captains_tests_50.take( 10 )
```

Out[35]:

```
[Captain(name='Smith  G C', country='South Africa', career='2002-2014',
matches=109, won=53, lost=29, ties=27, toss=58),
 Captain(name='Border  A R', country='Australia', career='1978-1994', m
atches=93, won=32, lost=22, ties=38, toss=47),
 Captain(name='Fleming  S P', country='New Zealand', career='1994-200
8', matches=80, won=28, lost=27, ties=25, toss=38),
 Captain(name='Ponting  R T', country='Australia', career='1995-2012',
matches=77, won=48, lost=16, ties=13, toss=37),
 Captain(name='Lloyd  C H', country='West Indies', career='1966-1984',
matches=74, won=36, lost=12, ties=26, toss=35),
 Captain(name='Dhoni  M S*', country='India', career='2005-', matches=6
0, won=27, lost=18, ties=15, toss=27),
 Captain(name='Waugh  S R', country='Australia', career='1985-2004', ma
tches=57, won=41, lost=9, ties=7, toss=31),
 Captain(name='Ranatunga  A', country='Sri Lanka', career='1982-2000',
matches=56, won=12, lost=19, ties=25, toss=29),
 Captain(name='Atherton  M A', country='England', career='1989-2001', m
atches=54, won=13, lost=21, ties=20, toss=23),
 Captain(name='Cronje  W J', country='South Africa', career='1992-200
0', matches=53, won=27, lost=11, ties=15, toss=22)]
```

In [36]:

```
# Sort the captains by percentage of wins
captain_top = captains_tests_50.map(
    lambda rec: ( rec.name,
                 round( rec.won/rec.matches,
                       2 ) ) ).sortBy( lambda rec: rec[1], ascending = False )
```

In [37]:

```
captain_top.collect()
```

Out[37]:

```
[('Waugh  S R', 0.72),
 ('Ponting  R T', 0.62),
 ('Cronje  W J', 0.51),
 ('Vaughan  M P', 0.51),
 ('Smith  G C', 0.49),
 ('Lloyd  C H', 0.49),
 ('Dhoni  M S*', 0.45),
 ('Fleming  S P', 0.35),
 ('Border  A R', 0.34),
 ('Atherton  M A', 0.24),
 ('Ranatunga  A', 0.21)]
```

Joining multiple data sets

In [38]:

```
# Lets join both ODI and Test captaincy details.
# Default is inner join...
all_time_best_captains = captains_100_percent_wins.join( captain_top )
```

In [39]:

```
all_time_best_captains.collect()
```

Out[39]:

```
[('Smith  G C', (0.61, 0.49)),
 ('Border  A R', (0.6, 0.34)),
 ('Fleming  S P', (0.45, 0.35)),
 ('Cronje  W J', (0.71, 0.51)),
 ('Ponting  R T', (0.72, 0.62)),
 ('Ranatunga  A', (0.46, 0.21)),
 ('Dhoni  M S*', (0.55, 0.45)),
 ('Waugh  S R', (0.63, 0.72))]
```

In [40]:

```
## Best by test match wins
all_time_best_captains.sortBy( lambda rec: rec[1][1],
                              ascending = False ).collect()
```

Out[40]:

```
[('Waugh  S R', (0.63, 0.72)),
 ('Ponting  R T', (0.72, 0.62)),
 ('Cronje  W J', (0.71, 0.51)),
 ('Smith  G C', (0.61, 0.49)),
 ('Dhoni  M S*', (0.55, 0.45)),
 ('Fleming  S P', (0.45, 0.35)),
 ('Border  A R', (0.6, 0.34)),
 ('Ranatunga  A', (0.46, 0.21))]
```

In [41]:

```
## Best by ODI match wins
all_time_best_captains.sortBy( lambda rec: rec[1][0],
                              ascending = False ).collect()
```

Out[41]:

```
[('Ponting  R T', (0.72, 0.62)),
 ('Cronje  W J', (0.71, 0.51)),
 ('Waugh  S R', (0.63, 0.72)),
 ('Smith  G C', (0.61, 0.49)),
 ('Border  A R', (0.6, 0.34)),
 ('Dhoni  M S*', (0.55, 0.45)),
 ('Ranatunga  A', (0.46, 0.21)),
 ('Fleming  S P', (0.45, 0.35))]
```

In [42]:

```
## Now let's flatten the structure and store the results into a file...
best_captains = all_time_best_captains.map( lambda rec:
                                           ( rec[0],
                                            rec[1][0],
                                            rec[1][1] ) )
best_captains.take( 10 )
```

Out[42]:

```
[('Smith  G C', 0.61, 0.49),
 ('Border  A R', 0.6, 0.34),
 ('Fleming  S P', 0.45, 0.35),
 ('Cronje  W J', 0.71, 0.51),
 ('Ponting  R T', 0.72, 0.62),
 ('Ranatunga  A', 0.46, 0.21),
 ('Dhoni  M S*', 0.55, 0.45),
 ('Waugh  S R', 0.63, 0.72)]
```

Saving results into filesystem

In [43]:

```
best_captains.saveAsTextFile( "file:///home/hadoop/lab/results/captains")
```

Check results

Go to you VM's putty terminal and go to the results directory

cd /home/hadoop/lab/results/captains/

cat part-0000*

Saving results into one file. Forcing the bring all results into one partition

In [44]:

```
## Consolidate into one partition
best_captains.repartition( 1 ).                                     \
    saveAsTextFile( "file:///home/hadoop/lab/results/bestcaptains")
```

Check results

Go to you VM's putty terminal and go to the results directory

cd /home/hadoop/lab/results/bestcaptains/

cat part-00000

Make note of lessons learnt in this lab