Advanced Analytics with Spark

이창언

Ch1. 빅데이터 분석하기

데이터 과학의 어려움

1. 성공적인 분석을 위한 작업의 대부분은 데이터의 전처리 과정에서 이루어진다.
2. 데이터 과학에서 반복(iteration)은 기본적인 과정이다.
3. 잘 돌아가는 모델이 완성되었다고 해서 끝이 아니다.

스파크를 활용해라

스파크는 탐색용 분석 시스템과
운영용 분석 시스템의 간격을 좁힐 수 있다.

Ch2. 스칼라와 스파크를 활용한 데이터 분석

```
import org.apache.spark.sql.{DataFrame, Dataset, Row, SparkSession}
import org.apache.spark.sql.functions._ // for lit(), first(), etc.

// Spark 2.0.2 API for Scala
// sparkSession : https://spark.apache.org/docs/2.0.2/api/scala/index.html#org.apache.spark.sql.SparkSession
// DataSet : https://spark.apache.org/docs/2.0.2/api/scala/index.html#org.apache.spark.sql.Dataset
```

```
import org.apache.spark.sql.{DataFrame, Dataset, Row, SparkSession}
import org.apache.spark.sql.functions._
```

```
case class MatchData(
  id_1: Int,
  id_2: Int,
  cmp_fname_c1: Option[Double],
  cmp_fname_c2: Option[Double],
  cmp_lname_c1: Option[Double],
  cmp_lname_c2: Option[Double],
  cmp_sex: Option[Int],
  cmp_bd: Option[Int],
  cmp_bm: Option[Int],
  cmp_by: Option[Int],
  cmp_plz: Option[Int],
  is_match: Boolean
)
```

```
defined class MatchData
```

Took 1 sec. Last updated by anonymous at January 24 2017, 11:18:43 AM.

```
val spark = SparkSession.builder
  .appName("Intro")
  .getOrCreate
import spark.implicits._ // DataFrame으로 변환을 위한 도움을 준다.
```

```
spark: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@2b930bce
import spark.implicits._
```

```
val preview = spark.read.csv("/Users/lee/Documents/sparks/Season2/Advanced_Analytics_With_Spark/linkage/block_1.csv")
    preview.show()
```

```
| id_1| id_2|     cmp_fname_c1|cmp_fname_c2|cmp_lname_c1|cmp_lname_c2|cmp_sex|cmp_bd|cmp_bm|cmp_by|cmp_plz|is_match|
|37291|53113|0.833333333333333|           ?|           1|           ?|      1|     1|     1|     1|      0|    TRUE|
|39086|47614|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
|70031|70237|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
|84795|97439|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
|36950|42116|                1|           ?|           1|           1|      1|     1|     1|     1|      1|    TRUE|
|42413|48491|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
|25965|64753|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
|49451|90407|                1|           ?|           1|           ?|      1|     1|     1|     1|      0|    TRUE|
|39932|40902|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
|46626|47940|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
|48948|98379|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
| 4767| 4826|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
|45463|69659|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
|11367|13169|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
|10782|89636|                1|           ?|           1|           ?|      1|     0|     1|     1|      1|    TRUE|
|26206|39147|                1|           ?|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
|16662|27083|                1|           1|           1|           ?|      1|     1|     1|     1|      1|    TRUE|
```

Took 2 sec. Last updated by anonymous at January 24 2017, 11:18:45 AM.

```
preview.schema.foreach(println)
```

```
StructField(_c0,StringType,true)
StructField(_c1,StringType,true)
StructField(_c2,StringType,true)
StructField(_c3,StringType,true)
StructField(_c4,StringType,true)
StructField(_c5,StringType,true)
StructField(_c6,StringType,true)
StructField(_c7,StringType,true)
StructField(_c8,StringType,true)
StructField(_c9,StringType,true)
StructField(_c10,StringType,true)
StructField(_c11,StringType,true)
```

Took 0 sec. Last updated by anonymous at January 24 2017, 11:18:45 AM.

```
val parsed = spark.read
      .option("header", "true") // 파일의 첫 줄을 필드 명으로 사용
      .option("nullValue", "?") // 필드 데이터를 변경( "?" => null )
      .option("inferSchema", "true") // 데이터 타입을 추론한다.
      .csv("/Users/lee/Documents/sparks/Season2/Advanced_Analytics_With_Spark/linkage/block_1.csv")
```

```
parsed: org.apache.spark.sql.DataFrame = [id_1: int, id_2: int ... 10 more fields]
```

Took 2 sec. Last updated by anonymous at January 24 2017, 11:18:47 AM.

```
parsed.show()
```

```
+-----+-----+-----------------+------------+------------+------------+-------+------+------+------+-------+--------+
| id_1| id_2|     cmp_fname_c1|cmp_fname_c2|cmp_lname_c1|cmp_lname_c2|cmp_sex|cmp_bd|cmp_bm|cmp_by|cmp_plz|is_match|
+-----+-----+-----------------+------------+------------+------------+-------+------+------+------+-------+--------+
|37291|53113|0.833333333333333|        null|         1.0|        null|      1|     1|     1|     1|      0|    true|
|39086|47614|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      1|    true|
|70031|70237|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      1|    true|
|84795|97439|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      1|    true|
|36950|42116|              1.0|        null|         1.0|         1.0|      1|     1|     1|     1|      1|    true|
|42413|48491|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      1|    true|
|25965|64753|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      1|    true|
|49451|90407|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      0|    true|
|39932|40902|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      1|    true|
|46626|47940|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      1|    true|
|48948|98379|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      1|    true|
| 4767| 4826|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      1|    true|
|45463|69659|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      1|    true|
|11367|13169|              1.0|        null|         1.0|        null|      1|     1|     1|     1|      1|    true|
|10782|89636|              1.0|        null|         1.0|        null|      1|     0|     1|     1|      1|    true|
```

```
val schema = parsed.schema
    schema.foreach(println)
```

FINISHED

```
schema: org.apache.spark.sql.types.StructType = StructType(StructField(id_1,IntegerType,true), StructField(id_2,IntegerType,true), StructField(cmp_fname_c1,DoubleType,
ue), StructField(cmp_fname_c2,DoubleType,true), StructField(cmp_lname_c1,DoubleType,true), StructField(cmp_lname_c2,DoubleType,true), StructField(cmp_sex,IntegerType,t
e), StructField(cmp_bd,IntegerType,true), StructField(cmp_bm,IntegerType,true), StructField(cmp_by,IntegerType,true), StructField(cmp_plz,IntegerType,true), StructFiel
(is_match,BooleanType,true))
StructField(id_1,IntegerType,true)
StructField(id_2,IntegerType,true)
StructField(cmp_fname_c1,DoubleType,true)
StructField(cmp_fname_c2,DoubleType,true)
StructField(cmp_lname_c1,DoubleType,true)
StructField(cmp_lname_c2,DoubleType,true)
StructField(cmp_sex,IntegerType,true)
StructField(cmp_bd,IntegerType,true)
StructField(cmp_bm,IntegerType,true)
StructField(cmp_by,IntegerType,true)
StructField(cmp_plz,IntegerType,true)
StructField(is_match,BooleanType,true)
```

```
parsed.count()
```

```
res57: Long = 574913
```

```
parsed.cache()
```

```
res59: parsed.type = [id_1: int, id_2: int ... 10 more fields]
```

```
parsed.groupBy("is_match").count().orderBy($"count".desc).show()
```

```
+--------+------+
|is_match| count|
+--------+------+
|   false|572820|
|    true|  2093|
+--------+------+
```

Took 1 sec. Last updated by anonymous at January 24 2017, 11:18:50 AM.

```
// ex)  parsed.describe("id_1","id_2") : 인자값을 넣어 해당 필드만 적용할 수 있다.
parsed.createOrReplaceTempView("parks")
    spark.sql("""
      SELECT is_match, COUNT(*) cnt
      FROM parks
      GROUP BY is_match
      ORDER BY cnt DESC
    """).show()
```

```
+--------+------+
|is_match|   cnt|
+--------+------+
|   false|572820|
|    true|  2093|
+--------+------+
```

Took 2 sec. Last updated by anonymous at January 24 2017, 11:18:51 AM.

FINISHED

```
// describe() : numeric columns, including count, mean, stddev, min, and max의 통계를 리턴해준다.
// ex)  parsed.describe("id_1","id_2") : 인자값을 넣어 해당 필드만 적용할 수 있다.
val summary = parsed.describe()
summary.show()
```

```
+-------+------------------+-----------------+------------------+------------------+------------------+------------------+------------------+--------------------
-------------+------------------+-------------------+
|summary|              id_1|             id_2|      cmp_fname_c1|      cmp_fname_c2|      cmp_lname_c1|      cmp_lname_c2|           cmp_sex|              cmp_bd
       cmp_bm|            cmp_by|            cmp_plz|
+-------+------------------+-----------------+------------------+------------------+------------------+------------------+------------------+--------------------
-------------+------------------+-------------------+
|  count|            574913|           574913|            574811|             10325|            574913|               239|            574913|              574851
       574851|            574851|             573618|
|   mean|33271.962171667714| 66564.6636865056| 0.7127592938251666|0.8977586763518972|0.3155724578099534|0.3269155414552904| 0.9550923357099248|0.2247556323290730
86361857246487|0.2226663952919974|0.005494946113964...|
| stddev|23622.669425933625|23642.00230967225|0.38892864524635457| 0.274257752043053|0.3342494687554249|  0.378309202054067|0.2071015224050438|0.4174216587235566
98712818281624|0.4160365041645614| 0.07392402321301918|
|    min|                 1|                6|               0.0|               0.0|               0.0|               0.0|                 0|                   0
            0|                 0|                  0|
|    max|             99894|           100000|               1.0|               1.0|               1.0|               1.0|                 1|                   1
            1|                 1|                  1|
+-------+------------------+-----------------+------------------+------------------+------------------+------------------+------------------+--------------------
-------------+------------------+-------------------+
summary.select("summary", "cmp_fname_c1", "cmp_fname_c2").show()
```

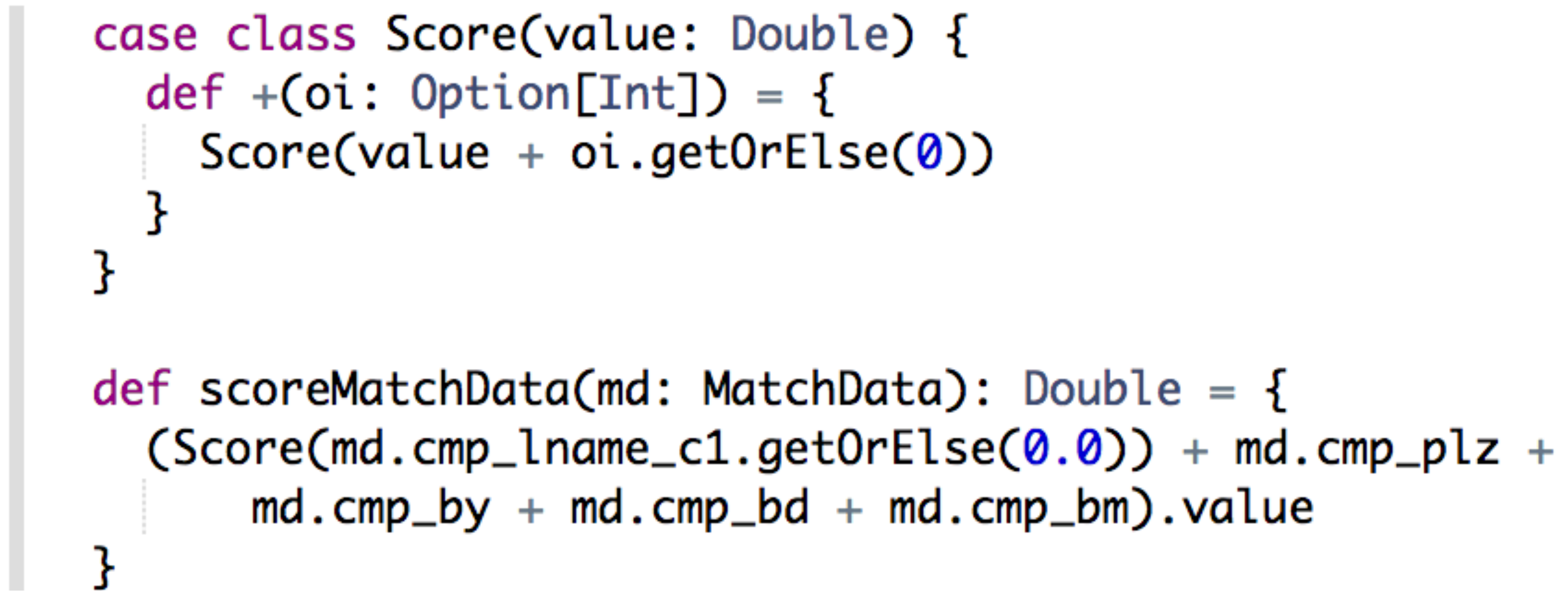
```
+-------+------------------+------------------+
|summary|      cmp_fname_c1|      cmp_fname_c2|
+-------+------------------+------------------+
|  count|            574811|             10325|
|   mean| 0.712759293825166|0.8977586763518972|
| stddev|0.3889286452463545| 0.274257752043053|
|    min|               0.0|               0.0|
|    max|               1.0|               1.0|
+-------+------------------+------------------+
```

Took 4 sec. Last updated by anonymous at January 24 2017, 11:18:55 AM.

```
val matches = parsed.where("is_match = true")
val misses = parsed.filter($"is_match" === lit(false))
val matchSummary = matches.describe()
val missSummary = misses.describe()
```

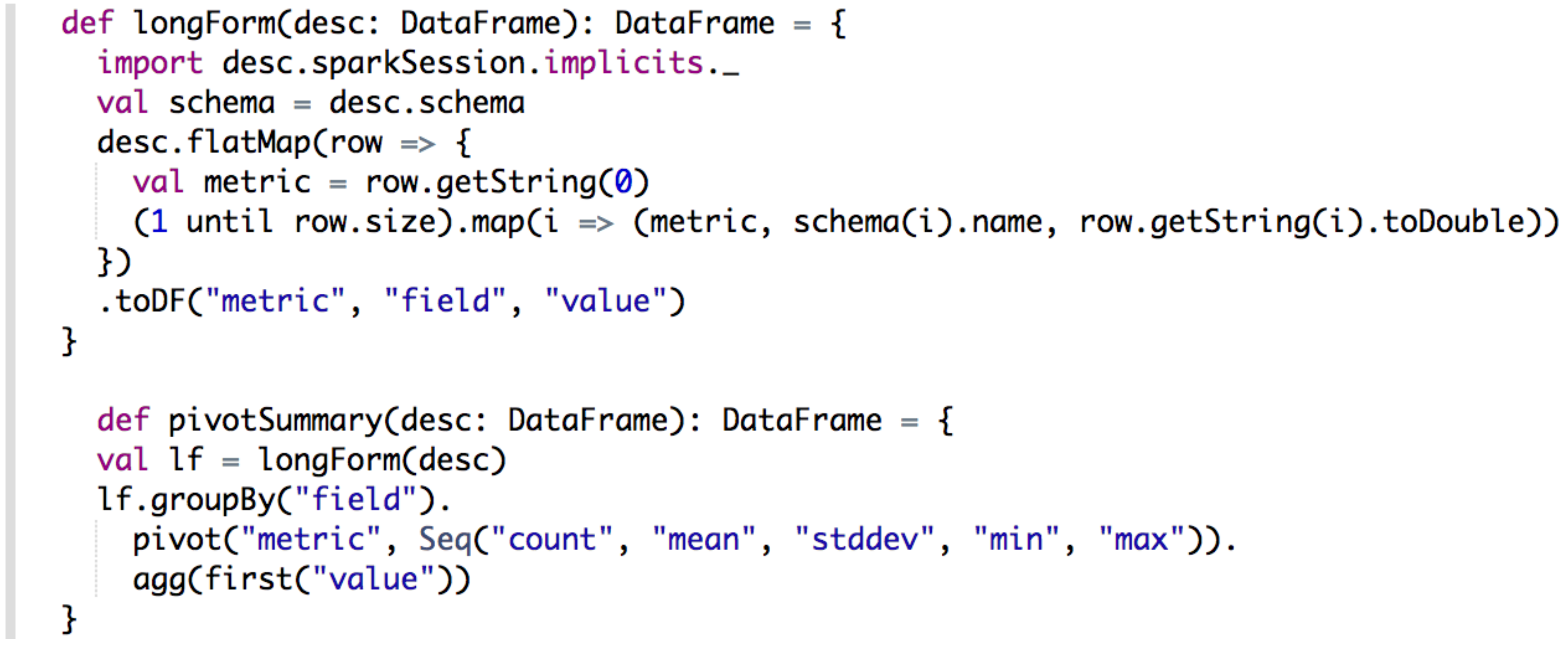
```
matches: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [id_1: int, id_2: int ... 10 more fields]
misses: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [id_1: int, id_2: int ... 10 more fields]
matchSummary: org.apache.spark.sql.DataFrame = [summary: string, id_1: string ... 10 more fields]
missSummary: org.apache.spark.sql.DataFrame = [summary: string, id_1: string ... 10 more fields]
```

Took 5 sec. Last updated by anonymous at January 24 2017, 11:19:00 AM.



```scala
case class Score(value: Double) {
  def +(oi: Option[Int]) = {
    Score(value + oi.getOrElse(0))
  }
}

def scoreMatchData(md: MatchData): Double = {
  (Score(md.cmp_lname_c1.getOrElse(0.0)) + md.cmp_plz +
      md.cmp_by + md.cmp_bd + md.cmp_bm).value
}
```



```scala
def longForm(desc: DataFrame): DataFrame = {
  import desc.sparkSession.implicits._
  val schema = desc.schema
  desc.flatMap(row => {
    val metric = row.getString(0)
    (1 until row.size).map(i => (metric, schema(i).name, row.getString(i).toDouble))
  })
  .toDF("metric", "field", "value")
}

  def pivotSummary(desc: DataFrame): DataFrame = {
  val lf = longForm(desc)
  lf.groupBy("field").
    pivot("metric", Seq("count", "mean", "stddev", "min", "max")).
    agg(first("value"))
}
```

```scala
val matchSummary = matches.describe()
val missSummary = misses.describe()
```

```
matchSummary: org.apache.spark.sql.DataFrame = [summary: string, id_1: string ... 10 more fields]
missSummary: org.apache.spark.sql.DataFrame = [summary: string, id_1: string ... 10 more fields]
```

Took 5 sec. Last updated by anonymous at January 24 2017, 11:19:07 AM. (outdated)

```scala
val matchSummaryT = pivotSummary(matchSummary)
val missSummaryT = pivotSummary(missSummary)
```

```
matchSummaryT: org.apache.spark.sql.DataFrame = [field: string, count: double ... 4 more fields]
missSummaryT: org.apache.spark.sql.DataFrame = [field: string, count: double ... 4 more fields]
```

Took 4 sec. Last updated by anonymous at January 24 2017, 11:19:07 AM. (outdated)

```scala
matchSummaryT.createOrReplaceTempView("match_desc")
missSummaryT.createOrReplaceTempView("miss_desc")
```

Took 1 sec. Last updated by anonymous at January 24 2017, 11:19:08 AM. (outdated)

```
spark.sql("""
  SELECT a.field, a.count + b.count total, a.mean - b.mean delta
  FROM match_desc a INNER JOIN miss_desc b ON a.field = b.field
  ORDER BY delta DESC, total DESC
""").show()
```

```
+------------+--------+-------------------+
|       field|   total|              delta|
+------------+--------+-------------------+
|        id_1|574913.0| 1173.1784091823356|
|     cmp_plz|573618.0| 0.9524975516429005|
|cmp_lname_c2|   239.0| 0.8136949970410103|
|      cmp_by|574851.0| 0.7763379425859384|
|      cmp_bd|574851.0| 0.7732820129086737|
|cmp_lname_c1|574913.0| 0.6844795197262346|
|      cmp_bm|574851.0|  0.510834819548174|
|cmp_fname_c1|574811.0|0.28531156828536086|
|cmp_fname_c2| 10325.0|0.09900440489032658|
|     cmp_sex|574913.0|0.03452211590529575|
|        id_2|574913.0|-15732.615614206828|
+------------+--------+-------------------+
```

Took 4 sec. Last updated by anonymous at January 24 2017, 11:19:12 AM.

```
val matchData = parsed.as[MatchData] // as() : 새로운 DataSet를 리턴한다.
val scored = matchData.map(md => {
  (scoreMatchData(md), md.is_match)
}).toDF("score", "is_match")
crossTabs(scored, 4.0).show()
```

```
matchData: org.apache.spark.sql.Dataset[MatchData] = [id_1: int, id_2: int ... 10 more fields]
scored: org.apache.spark.sql.DataFrame = [score: double, is_match: boolean]
+-----+----+------+
|above|true| false|
+-----+----+------+
| true|2087|    66|
|false|   6|572754|
+-----+----+------+
```

Took 6 sec. Last updated by anonymous at January 24 2017, 11:19:15 AM.

Q & A