2018-2
Spark Project

김조현

RDD

collect, filter

train data 읽기

```
act = sc.textFile('file:///home/ubuntu/18-2E/sparkProject/train_activity.csv').map(lambda line : line.split(','))
label = sc.textFile('file:///home/ubuntu/18-2E/sparkProject/train_label.csv').map(lambda line : line.split(','))
```

pandas로 확인하기 ¶

```
pd_act = pd.DataFrame(act.collect())
pd_act.head()
```

	0	1	2	3	4
0	wk	acc_id	cnt_dt	play_time	npc_exp
1	7	3dc6f2875dc6e6f35b9e2bdb25b391a8003386ff23becd...	4	2.08881416107027	4.4050571352657
2	8	3dc6f2875dc6e6f35b9e2bdb25b391a8003386ff23becd...	5	2.67346049372266	4.76017781944869
3	3	b8856358ff62e596fa07e3e40b8e7fd4b7729263c72b44...	2	-0.649521652982493	-0.231020534896592
4	4	b8856358ff62e596fa07e3e40b8e7fd4b7729263c72b44...	2	-0.65823531742848	-0.231874043837118

```
# column 정보 제거
act = act.filter(lambda line : line[0] != "wk")
label = label.filter(lambda line: line[0] != "acc_id")
```

→ (header = true)

각 주차에 한번이라도 활동한 사람 수

wk : 활동 주 (1~8)

groupBy, mapValues, map, sortBy, reduceByKey

```
result1 = act.groupBy(lambda x : x[0])\
    .mapValues(lambda x : len(x))\
    .sortBy(lambda x: x[1], False)
result1.collect()
```

```
[('8', 100000),
 ('7', 62838),
 ('5', 52585),
 ('4', 51430),
 ('6', 50548),
 ('3', 46122),
 ('2', 43093),
 ('1', 33707)]
```

→
시간이 지남에 따라
8주차에 가까워 질수록
활동 인원이 증가함

각 주차별 모든 사람의 접속일수 합

cnt_dt : 해당 주에 한번 이상 접속한 일수

```
result2 = act.map(lambda x : (int(x[0]), int(x[2])))\
    .reduceByKey(lambda a, b : a+b)\
    .sortBy(lambda x : x[1], False)
result2.collect()
```

```
[(8, 405036),
 (7, 282452),
 (5, 241248),
 (4, 235921),
 (3, 221524),
 (6, 214319),
 (2, 181008),
 (1, 165063)]
```

(주차, 접속일수) pair을 만들어서
각 주차 별 총 접속일수를
reduceByKey 로 구함

→
6주차의 경우
접속 빈도가 낮은 경향을 보임

Spark SQL

Head, take, describe, show, printSchema

```
source = "file:///home/ubuntu/18-2E/sparkProject/train_activity.csv"
df = spark.read.csv(source, header = "true")
```

header = true

```
df.head()
```

```
df.take(2)
```

```
df.describe("wk").show()
```

```
+-------+-----------------+
|summary|               wk|
+-------+-----------------+
|  count|           440323|
|   mean|5.155438166981965|
| stddev| 2.31232206450818|
|    min|                1|
|    max|                8|
+-------+-----------------+
```

```
df.printSchema()
```

```
root
 |-- wk: string (nullable = true)
 |-- acc_id: string (nullable = true)
 |-- cnt_dt: string (nullable = true)
 |-- play_time: string (nullable = true)
 |-- npc_exp: string (nullable = true)
 |-- npc_hongmun: string (nullable = true)
 |-- quest_exp: string (nullable = true)
 |-- quest_hongmun: string (nullable = true)
 |-- item_hongmun: string (nullable = true)
 |-- game_combat_time: string (nullable = true)
 |-- get_money: string (nullable = true)
```

Sql, select, alias

sql문을 직접 이용한 연산

```
df.createOrReplaceTempView("act")
spark.sql("SELECT wk, acc_id, cnt_dt FROM act WHERE wk > 6").show(5)
```

```
+---+--------------------+------+
| wk|              acc_id|cnt_dt|
+---+--------------------+------+
|  7|3dc6f2875dc6e6f35...|     4|
|  8|3dc6f2875dc6e6f35...|     5|
|  7|b8856358ff62e596f...|     2|
|  8|b8856358ff62e596f...|     5|
|  8|fa883ca7505082114...|     6|
+---+--------------------+------+
```

→
주차가 6보다 큰 경우
wk, acc_id, cnt_dt 출력

```
heavy = spark.sparkContext.broadcast([5,6,7])
df.select(df['acc_id'],df['wk'],df['cnt_dt'], df['cnt_dt'].isin(heavy.value).alias("heavyUser")).show(10)
```

```
+--------------------+---+------+---------+
|              acc_id| wk|cnt_dt|heavyUser|
+--------------------+---+------+---------+
|3dc6f2875dc6e6f35...|  7|     4|    false|
|3dc6f2875dc6e6f35...|  8|     5|     true|
|b8856358ff62e596f...|  3|     2|    false|
|b8856358ff62e596f...|  4|     2|    false|
|b8856358ff62e596f...|  5|     4|    false|
|b8856358ff62e596f...|  7|     2|    false|
|b8856358ff62e596f...|  8|     5|     true|
|fa883ca7505082114...|  8|     6|     true|
|d094b6b1c5d0a147e...|  8|     3|    false|
|38e7088d64485baba...|  1|     6|     true|
+--------------------+---+------+---------+
only showing top 10 rows
```

→
각 사람이 그 주에 5번 이상 게임에서 활동했다면

heavyUser !

when, otherwise, alias, mean, collect_set, countDistinct

```
from pyspark.sql import functions

col = functions.when(df.cnt_dt >= 5 , "heavy").otherwise("light").alias("userType")
df.select(df.acc_id, df.wk, df.cnt_dt, col).show(10)
```

```
+--------------------+---+------+--------+
|              acc_id| wk|cnt_dt|userType|
+--------------------+---+------+--------+
|3dc6f2875dc6e6f35...|  7|     4|   light|
|3dc6f2875dc6e6f35...|  8|     5|   heavy|
|b8856358ff62e596f...|  3|     2|   light|
|b8856358ff62e596f...|  4|     2|   light|
|b8856358ff62e596f...|  5|     4|   light|
|b8856358ff62e596f...|  7|     2|   light|
|b8856358ff62e596f...|  8|     5|   heavy|
|fa883ca7505082114...|  8|     6|   heavy|
|d094b6b1c5d0a147e...|  8|     3|   light|
|38e7088d64485baba...|  1|     6|   heavy|
+--------------------+---+------+--------+
```

↑
각 사람이 그 주에 5번 이상 게임에서 활동했다면

userType = heavy

↓
전체 사람의 평균 일주일 활동 일 수

주차는 8주차까지

```
df.select(functions.mean(df['cnt_dt'])).show()
```

```
+-----------------+
|      avg(cnt_dt)|
+-----------------+
|4.4207797457777532|
+-----------------+
```

```
df.select(functions.collect_set('wk')).show()
```

```
+--------------------+
|     collect_set(wk)|
+--------------------+
|[3, 1, 2, 5, 8, 4...|
+--------------------+
```

```
df.select(functions.countDistinct('wk')).show()
```

```
+------------------+
|count(DISTINCT wk)|
+------------------+
|                 8|
+------------------+
```

Spark ML

Decision Tree

```
# 1. 스파크세션 생성
spark = SparkSession ₩
    .builder ₩
    .appName("decision_tree_userType") ₩
    .master("local[*]") ₩
    .getOrCreate()
```

```
def isHeavyUser(cnt_dt):
    if cnt_dt >= 5:
        return 1.0
    else:
        return 0.0
```

```
# Label(heavyUser:1.0, lightUser:0.0)
isHU = functions.udf(lambda cnt_dt: isHeavyUser(cnt_dt))
```

```
d1 = spark.read.option("header", "true") ₩
    .option("sep", ",").option("inferSchema", True) ₩
    .option("mode", "DROPMALFORMED") ₩
    .csv("file:///home/ubuntu/18-2E/sparkProject/train_activity.csv")

d2 = d1.toDF('wk', 'acc_id', 'cnt_dt', 'play_time', 'npc_exp', 'npc_hongmun',
       'party_chat', 'guild_chat', 'faction_chat', 'cnt_use_buffitem',
       'gathering_cnt', 'making_cnt')

d2.printSchema()
```

```
root
 |-- wk: integer (nullable = true)
 |-- acc_id: string (nullable = true)
 |-- cnt_dt: integer (nullable = true)
 |-- play_time: double (nullable = true)
```

```
# 사람별 평균 cnt_dt (주당 방문 일수)
d3 = d2.groupBy("acc_id").agg(functions.round(functions.avg("cnt_dt"), 1).alias("mean(cnt_dt)"))
d4 = d2.join(d3, ["acc_id"])
d4.select(d4["acc_id"], d4["mean(cnt_dt)"]).show(20, False)
```

```
+----------------------------------------------------------------+------------+
|acc_id                                                          |mean(cnt_dt)|
+----------------------------------------------------------------+------------+
|00446675fab526fc7b768e18ed051e3b5e341d5078fd2508c9c03f5258a2389a|3.7         |
|00446675fab526fc7b768e18ed051e3b5e341d5078fd2508c9c03f5258a2389a|3.7         |
|00446675fab526fc7b768e18ed051e3b5e341d5078fd2508c9c03f5258a2389a|3.7         |
|0148a24b0c6ea3da5f03ac5f516fe030d63fb88d222f1cd417073c7bb7edd71e|1.0         |
|02a4d8afc1c0359a3c0d28e3cd55cd8956ba02af055260a50c113b91e91e4573|2.0         |
|02a4d8afc1c0359a3c0d28e3cd55cd8956ba02af055260a50c113b91e91e4573|2.0         |
```

→
한 사람이 여러 번 나오기는 하지만
각 사람별로 평균 cnt_dt 값이 구해짐

```
# label 부여
d5 = d4.withColumn("isHeavyUser", isHU(d4["mean(cnt_dt)"]).cast("double"))
d5.select("acc_id", "mean(cnt_dt)", "isHeavyUser").show(20, False)
```

```
+----------------------------------------------------------------+------------+-----------+
|acc_id                                                          |mean(cnt_dt)|isHeavyUser|
+----------------------------------------------------------------+------------+-----------+
|00446675fab526fc7b768e18ed051e3b5e341d5078fd2508c9c03f5258a2389a|3.7         |0.0        |
|00446675fab526fc7b768e18ed051e3b5e341d5078fd2508c9c03f5258a2389a|3.7         |0.0        |
|00446675fab526fc7b768e18ed051e3b5e341d5078fd2508c9c03f5258a2389a|3.7         |0.0        |
|0148a24b0c6ea3da5f03ac5f516fe030d63fb88d222f1cd417073c7bb7edd71e|1.0         |0.0        |
|02a4d8afc1c0359a3c0d28e3cd55cd8956ba02af055260a50c113b91e91e4573|2.0         |0.0        |
|02a4d8afc1c0359a3c0d28e3cd55cd8956ba02af055260a50c113b91e91e4573|2.0         |0.0        |
|033611b3c479b8d62fb04b2377378f46d8017170ef758ce3f594b04934de7f4d|6.5         |1.0        |
```

→
mean(cnt_dt) 값으로
isHeavyUser 값
1, 0으로 라벨링

```
dataArr = d5.randomSplit([0.7, 0.3])
train = dataArr[0]
test = dataArr[1]
```

```
# HeavyUser 판단에 사용된 cnt_dt와 mean(cnt_dt) 제외
indexer = StringIndexer(inputCol="acc_id", outputCol="id_code")
assembler = VectorAssembler(inputCols=['wk', 'play_time', 'npc_exp', 'npc_hongmun',
                                       ⋮
       'gathering_cnt', 'making_cnt'], outputCol="features")
dt = DecisionTreeClassifier(labelCol="isHeavyUser", featuresCol="features").setMaxBins(40)

pipeline = Pipeline(stages=[indexer, assembler, dt])
model = pipeline.fit(train)

predict = model.transform(test)
predict.select("probability", "prediction", "isHeavyUser").show(3, False)
```

```
+----------------------------------------+----------+-----------+
|probability                             |prediction|isHeavyUser|
+----------------------------------------+----------+-----------+
|[0.032476357664646925,0.967523642335353]|1.0       |1.0        |
|[0.14059775840597757,0.8594022415940225]|1.0       |1.0        |
|[0.14059775840597757,0.8594022415940225]|1.0       |1.0        |
+----------------------------------------+----------+-----------+
```

```
evaluator = BinaryClassificationEvaluator(labelCol="isHeavyUser", metricName="areaUnderROC")
```

```
print(evaluator.evaluate(predict))
treeModel = model.stages[2]
print("Learned classification tree model:%s" % treeModel.toDebugString)
```

```
0.7030415516310541
Learned classification tree model:DecisionTreeClassificationModel (uid=DecisionTreeClassifier_41559485db47bbb05
  If (feature 8 <= -0.0234785876090536)
   If (feature 28 <= -0.188697394931091)
    If (feature 1 <= -0.6564500547355165)
```

끝