Table of contents

Spark MLlib provides

- An itemset mining algorithm based on the FP-growth algorithm, that extracts all the sets of items (of any length) with a minimum frequency;
- A rule mining algorithm, that extracts the association rules with a minimum frequency and a minimum confidence; notice that only the rules with one single item in the consequent of the rules are extracted.

The input dataset in this case is a set of transactions, where each transaction is defined as a set of items

A transactional dataset example

```
ABCD
AB
BC
ADE
```

It contains 4 transactions, and the distinct items are A, B, C, D, E.

1 The FP-Growth algorithm and Association rule mining

FP-growth is one of the most popular and efficient itemset mining algorithms. It is characterized by one single parameter: the minimum support threshold (**minsup**), that is the minimum frequency of the itemset in the input transational dataset; it can assume a real value in the range $(0, 1]$. The minsup threshold is used to limit the number of mined itemsets.

The input dataset is a transactional dataset.

Given a set of frequent itemsets, the frequent association rules can be mined. An association rule is mined if

- Its frequency is greater than the minimum support threshold **minsup** (i.e., a minimum frequency). The minsup value is specified during the itemset mining step and not during the association rule mining step.
- Its confidence is greater than the minimum confidence threshold **minconf** (i.e., a minimum correlation). It is a real value in the range $[0, 1]$.

The MLlib implementation of FP-growth is based on DataFrames, but differently from the other algorithms, the FP-growth algorithm is not invoked by using pipelines.

1.1 Steps for itemset and association rule mining in Spark

1. Instantiate an FP-Growth object
2. Invoke the `fit(input data)` method on the FP-Growth object
3. Retrieve the sets of frequent itemset and association rules by invoking the following methods of on the FP-Growth object
    - `freqItemsets()`
    - `associationRules()`

1.2 Input

The input of the MLlib itemset and rule mining algorithm is a DataFrame containing a column called `items`, whose data type is array of values. Each record of the input DataFrame contains one transaction (i.e., a set of items).

Example

Example of input data

```
transactions
ABCD
AB
BC
ADE
```

The column items must be created before invoking FP-growth

items
$[A, B, C, D]$
$[A, B]$
$[B, C]$
$[A, D, E]$

Each input line is stored in an array of strings. The generated DataFrame contains a column called items, which is an `ArrayType`, containing the lists of items associated with the input transactions.

Note

This example shows how to extract the set of frequent itemsets from a transactional dataset and the association rules from the extracted frequent itemsets.
The input dataset is a transactional dataset: each line of the input file contains a transaction (i.e., a set of items)

```
transactions
ABCD
AB
```

```
BC
ADE
```

```python
from pyspark.ml.fpm import FPGrowth
from pyspark.ml import Pipeline
from pyspark.ml import PipelineModel
from pyspark.sql.functions import col, split

# input and output folders
transactionsData = "ex_dataitemsets/transactions.csv"
outputPathItemsets = "Itemsets/"
outputPathRules = "Rules/"

# Create a DataFrame from transactions.csv
transactionsDataDF = spark.read.load(
    transactionsData,
    format="csv",
    header=True,
    inferSchema=True
)

# Transform Column transactions into an ArrayType
trsDataDF = transactionsDataDF \
    .selectExpr('split(transactions, " ")') \
    .withColumnRenamed("split(transactions, )", "items") # <1>

# Transform Column transactions into an ArrayType
trsDataDF = transactionsDataDF\
.selectExpr('split(transactions, " ")')\
.withColumnRenamed("split(transactions, )", "items")

# Create an FP-growth Estimator
fpGrowth = FPGrowth(
    itemsCol="items",
    minSupport=0.5,
    minConfidence=0.6
)

# Extract itemsets and rules
model = fpGrowth.fit(trsDataDF)

# Retrieve the DataFrame associated with the frequent itemsets
dfItemsets = model.freqItemsets

# Retrieve the DataFrame associated with the frequent rules
dfRules = model.associationRules

# Save the result in an HDFS output folder
dfItemsets.write.json(outputPathItemsets) # <2>

# Save the result in an HDFS output folder
dfRules.write.json(outputPathRules)
```

1. `'split(transactions, " ")'` is the `pyspark.sql.functions.split()` function. It returns a `SQL ArrayType`.
2. The result is stored in a JSON file because itemsets and rules are stored in columns associated with the data type Array. Hence, CSV files cannot be used to store the result.