FP-growth笔记

浮生的笔记 1月5日

FP-growth（Frequent Pattern Growth）模型，用于挖掘频繁项集和关联规则的算法，最近在学习这个算法，做一下笔记。

频繁项集

啤酒与尿布就是一个典型的例子。有研究发现，在超市的订单记录中，啤酒和尿布总是频繁共同出现在同一条订单记录里。这里啤酒和尿布就是频繁项集 。

关联规则

关联规则是在频繁项集的基础上得到的。关联规则指由集合 A，可以在某置信度下推出集合 B。通俗来说，就是如果 A 发生了，那么 B 也很有可能会发生。举个例子，有关联规则如：{'鸡蛋', '面包'} -> {'牛奶'}，该规则的置信度是 0.9，意味着在所有买了鸡蛋和面包的客户中，有90%的客户还买了牛奶。这里涉及到两个概念：支持度和置信度。

支持度

支持度指某频繁项集在整个数据集中的比例。假设数据集有 10 条记录，包含{'鸡蛋', '面包'}的有 5 条记录，那么{'鸡蛋', '面包'}的支持度就是 5/10 = 0.5。

置信度

置信度是针对某个关联规则定义的。有关联规则如{'鸡蛋', '面包'} -> {'牛奶'}，它的置信度计算公式为{'鸡蛋', '面包', '牛奶'}的支持度/{'鸡蛋', '面包'}的支持度。假设{'鸡蛋', '面包', '牛奶'}的支持度为 0.45，{'鸡蛋', '面包'}的支持度为 0.5，则{'鸡蛋', '面包'} -> {'牛奶'}的置信度为 0.45 / 0.5 = 0.9。

FP-growth

假设事务数据库如下

1	A,B,E
2	B,D
3	B,C
4	A,B,D
5	A,C
6	B,C

7	A,C
8	A,B,C,E
9	A,B,C

首先扫描一次数据库得到项目集

A	B	C	D	E
6	7	6	2	2

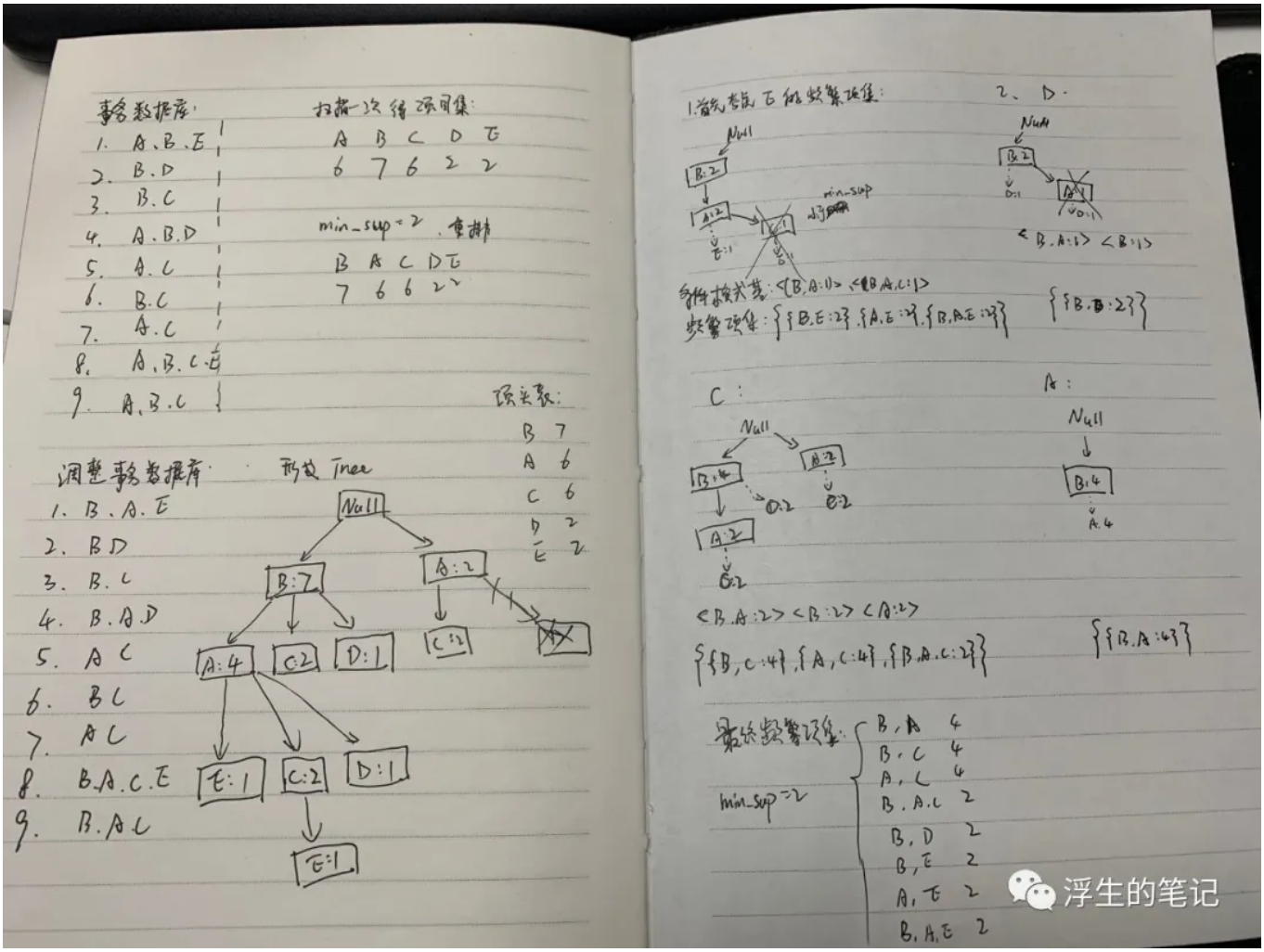
令最小支持度为2：min_sup=2,
过滤支持度小于2的元素，并对项目集内元素按频数重排,得到项头表

B	A	C	D	E
7	6	6	2	2

按频数调整事务数据库

1	B,A,E
2	B,D
3	B,C
4	B,A,D
5	A,C
6	B,C
7	A,C
8	B,A,C,E
9	B,A,C

按照调整后的数据库 形成FP-tree
形成FP-tree后，按照项头表的顺序，从低到高依此求得各个元素的频繁项集
画图比较麻烦，见下图笔记…



Scala实现

```
def main(args: Array[String]): Unit = {
    val appName = "FP-Growth关联规则"
    val sparkConfig = new SparkConf().setAppName(appName).setMaster("local[*]
    val spark: SparkSession = SparkSession.builder().config(sparkConfig).getO
    var sqlContext = spark.sqlContext
    //调整开发输出的日志信息
    val sc = spark.sparkContext
    sc.setLogLevel("ERROR")

    import spark.implicits._
    val dataList: List[(String)] = List(
      ("A,B,E"),
      ("B,D"),
      ("B,C"),
      ("A,B,D"),
      ("A,C"),
```

```
      ("B,C"),
      ("A,C"),
      ("A,B,C,E"),
      ("A,B,C")
    )
    val data = dataList.map(t => t.split(",")).toDF("items")

    val fpgrowth = new FPGrowth().setItemsCol("items").setMinSupport(0.2).set
    val model = fpgrowth.fit(data)

    // Display frequent itemsets.
    model.freqItemsets.show()

    // Display generated association rules.
    model.associationRules.show()

    // transform examines the input items against all the association rules a
    // consequents as prediction
    model.transform(data).show()
  }
```

Python实现-pyspark

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# @Time    : 2021-01-05 17:52
# @File    : FP-Growth.py
# @Author  : FS


from pyspark.ml.fpm import FPGrowth
from pyspark.sql import SparkSession
from pyspark.sql import functions as F

spark = SparkSession.builder.appName("FP-Growth关联规则").getOrCreate()
df = spark.createDataFrame([
    (0, 'A,B,E'),
    (1, 'B,D'),
```

```
    (2, 'B,C'),
    (3, 'A,B,D'),
    (4, 'A,C'),
    (5, 'B,C'),
    (6, 'A,C'),
    (7, 'A,B,C,E'),
    (8, 'A,B,C')
], ["id", "items"])
df = df.withColumn("items", F.split(df.items, ","))
df.show()

fpGrowth = FPGrowth(itemsCol="items", minSupport=0.2, minConfidence=0.5)
model = fpGrowth.fit(df)

# Display frequent itemsets.
model.freqItemsets.show()

# Display generated association rules.
model.associationRules.show()

# transform examines the input items against all the association rules and su
# consequents as prediction
model.transform(df).show()
```

Python实现-pyfpgrowth

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# @Time    : 2021-01-05 17:52
# @File    : FP-Growth.py
# @Author  : FS


import pandas as pd
import pyfpgrowth

df = pd.DataFrame([
    'A,B,E',
```

```
    'B,D',
    'B,C',
    'A,B,D',
    'A,C',
    'B,C',
    'A,C',
    'A,B,C,E',
    'A,B,C'
])
df.columns = ['items']
df['items'] = df['items'].map(lambda x: x.split(','))

res = []
for i in df.values.tolist():
    res.append(i[0])

patterns = pyfpgrowth.find_frequent_patterns(res, 2)
print(patterns)

rules = pyfpgrowth.generate_association_rules(patterns, 0.7)
print(rules)
```