商品关键词提取(1)

根据sku_detail表，提取商品的关键词

对电商产品提取关键词，参照对文本类型数据提取关键词的方式，将所有电商产品的所有详细描述组合成一个文本，提取关键词，提取技术：tfidf、textrank等

延申学习：jieba分词的使用

''' 内置函数open与codecs.open的区别： Python3直接用open。 Python2.x下用codecs.open，特别是有中文的情况，然后也可以避免踩到2.6下面io.open的坑。 如果希望代码同时兼容Python2和Python3，那么推荐用codecs.open '''

切割词；对切割之后的词语进行过滤，去除停用词，保留名词、英文词(长度大于2)和自定义词库中的词

分词

注意：hdfs的每台机器上的相同路径都要有同样的**词典和停用词**

In [6]:
```
# =====该cell跳过=====
# 用于查看strip()返回的结果，返回了str类型
[i.strip() for i in codecs.open(stopwords_path).readlines()]
codecs.open(stopwords_path).readlines()
```

In [3]:

```
import os
import jieba
import jieba.posseg as pseg
import codecs

abspath = "/root/workspace/3.rs_project/project2/notebook"

stopwords_path = os.path.join(abspath, 'keywordExtract/extract/baidu_stopwords.txt')

# 结巴加载用户词典
userDict_path = os.path.join(abspath, "keywordExtract/extract/词典/all.txt")
jieba.load_userdict(userDict_path)

# 停用词文本
stopwords_path = os.path.join(abspath, "keywordExtract/extract/baidu_stopwords.txt")


def get_stopwords_list():
    """返回stopwords列表"""
    stopwords_list = [i.strip()
                      for i in codecs.open(stopwords_path).readlines()]
    return stopwords_list

# 所有的停用词列表
stopwords_list = get_stopwords_list()

# 分词并过滤，定义过滤的规则
def cut_sentence(sentence):
    # print(sentence,"*"*100)
    # eg:[pair('今天', 't'), pair('有', 'd'), pair('雾', 'n'), pair('霾', 'g')]
    seg_list = pseg.lcut(sentence)
    seg_list = [i for i in seg_list if i.flag not in stopwords_list]
    filtered_words_list = []
    for seg in seg_list:
        # print(seg)
        if len(seg.word) <= 1:
            continue
        elif seg.flag == "eng":
            if len(seg.word) <= 2:
                continue
            else:
                filtered_words_list.append(seg.word)
        elif seg.flag.startswith("n"):
            filtered_words_list.append(seg.word)
        elif seg.flag in ["x", "eng"]:  # 是自定一个词语或者是英文单词
            filtered_words_list.append(seg.word)
    return filtered_words_list
```

In [2]:
```
# 显示分词效果
sentence = '''
攀升（IPASON）H68 i7 8700K/Z370/GTX1070Ti 8G/16G DDR4水冷游戏台式DIY组装电脑京东自营游
'''
cut_sentence(sentence)
```

...

In [7]:
```
import os
# 配置pyspark和spark driver运行时 使用的python解释器
JAVA_HOME = '/root/bigdata/jdk'
PYSPARK_PYTHON = '/miniconda2/envs/py365/bin/python'
# 当存在多个版本时，不指定很可能会导致出错
os.environ['PYSPARK_PYTHON'] = PYSPARK_PYTHON
os.environ['PYSPARK_DRIVER_PYTHON'] = PYSPARK_PYTHON
os.environ['JAVA_HOME'] = JAVA_HOME
# 配置spark信息
from pyspark import SparkConf
from pyspark.sql import SparkSession

SPARK_APP_NAME = "extractSKUKeyword"
SPARK_URL = "spark://192.168.58.100:7077"

conf = SparkConf()    # 创建spark config对象
config = (
    ("spark.app.name", SPARK_APP_NAME),    # 设置启动的spark的app名称，没有提供，将随
    ("spark.executor.memory", "2g"),    # 设置该app启动时占用的内存用量，默认1g，指一
    ("spark.master", SPARK_URL),    # spark master的地址
    ("spark.executor.cores", "2"),    # 设置spark executor使用的CPU核心数，指一台虚拟
    ("hive.metastore.uris", "thrift://localhost:9083"),    # 配置hive元数据的访问，否
    # 以下三项配置，可以控制执行器数量
#     ("spark.dynamicAllocation.enabled", True),
#     ("spark.dynamicAllocation.initialExecutors", 1),    # 1个执行器
#     ("spark.shuffle.service.enabled", True)
#   ('spark.sql.pivotMaxValues', '99999'),  # 当需要pivot DF，且值很多时，需要修改，默
)
# 查看更详细配置及说明：https://spark.apache.org/docs/latest/configuration.html

conf.setAll(config)

# 利用config对象，创建spark session
spark = SparkSession.builder.config(conf=conf).enableHiveSupport().getOrCreate()
```

2.2.1商品归类

对于电商来说，同一个关键词在不同品类数据之间通常都具有不同的含义，往往需要根据品类特性，分别解析提取关键词。

- 比如电子产品中的“苹果”和生鲜产品中的“苹果”，意义显然是不同的，不能相提并论。
- 比如用户正在浏览图书，那么相关的物品推荐，通常是不会出现食品的

正因如此，当我们进行商品关键词提取等处理时，都是要根据不同的类别进行独立的处理

但如何具体给商品归类，其实不是由我们随意决定的，而是**由产品设计等人员针对产品特性来划分**，以下划分仅供参考：

- 电子产品: 手机、相机、数码、电脑、办公 ==> 1-5
- 家居产品：家用电器、家居、家具、家装、厨具 ==> 6-10
- 服饰产品：男装、女装、童装、内衣、女鞋、箱包、钟表、珠宝、男鞋、运动、户外 ==> 11-21
- 资产产品：房产、汽车、汽车用品 ==> 22-24
- 母婴用品：母婴、玩具乐器 ==> 25-26
- 食用产品：食品、酒类、生鲜、特产 ==> 27-30
- 影音图书产品：图书、音像、电子书 ==> 31-33
- 旅游出行产品：机票、酒店、旅游、生活 ==> 34-37

所以这里我们需要对这几大类别的商品，分别进行关键词的提取工作

In [5]:
```
# 查看所有的一级分类
spark.sql('select * from tb_goods_category where parent_id is null').show(100)
```

...

In [8]:
```
# 首先处理电子产品
electronic_product = spark.sql('select * from sku_detail where category1_id >0 and ca
electronic_product.show()
```

...

2.2.2数据整合

为了让数据表现出足够的特征，这里我们把一个商品所有的详细信息都拼接为一个长文本字符串

In [9]:
```
# 所有文本字段 拼接成文本  有以下两种方法
# 第一种：写sql语句
sql1='''
select goods_id,sku_id,category1_id,category2_id,category3_id,\
    concat_ws(
        ',',name,caption,price,category1,category1,category3,specification
    ) summary from sku_detail
'''

# spark.sql(sql1).sort('sku_id').select('summary').show(1,truncate=False)
# 第二种：在spark sql - dataframe上操作
from pyspark.sql.functions import concat_ws
ret = electronic_product.select("goods_id", "sku_id", "category1_id", "category2_id",
    concat_ws(',',electronic_product.category1,\
              electronic_product.category2,\
              electronic_product.category3,\
              electronic_product.name,\
               electronic_product.price,\
               electronic_product.caption,\
               electronic_product.specification
              ).alias('summary')
)
ret.show(10)
# ret.sort('sku_id').select('summary').show(1,truncate=False)
```

```
+--------+------+-----------+-----------+-----------+--------------------+
|goods_id|sku_id|category1_id|category2_id|category3_id|             summary|
+--------+------+-----------+-----------+-----------+--------------------+
|     135|   148|          3|         41|        140|数码,数码配件,读卡器,随身厅
W...|
|     451|   463|          3|         41|        140|数码,数码配件,读卡器,飞花令
安...|
|     458|   471|          3|         41|        140|数码,数码配件,读卡器,【包邮】
飞...|
|     483|   496|          3|         41|        140|数码,数码配件,读卡器,品胜（P
I...|
|     820|   833|          3|         41|        140|数码,数码配件,读卡器,LEXAR...|
|    1075|  1088|          2|         40|        135|相机,摄影摄像,数码相框,青美
壁...|
|    1225|  1238|          3|         41|        140|数码,数码配件,读卡器,dypla...|
|    1329|  1342|          3|         41|        140|数码,数码配件,读卡器,绿联（U
G...|
|    1567|  1580|          2|         40|        135|相机,摄影摄像,数码相框,HNM ...|
|    1578|  1591|          3|         41|        140|数码,数码配件,读卡器,kisdi...|
+--------+------+-----------+-----------+-----------+--------------------+
only showing top 10 rows
```

2.2.3基于TextRank提取关键词

使用jieba中文分词自带的textrank方法进行处理

使用示范：

In [37]:
```
1 ret.where('category1_id=4').first()
```

Out[37]:
```
Row(goods_id=29588, sku_id=29601, category1_id=4, category2_id=45, category3_id=158, summary='电脑,电脑整机,游戏本,戴尔DELL灵越游匣Master15.6英寸游戏笔记本电脑(i5-7300HQ 8G 128GSSD+1T GTX1050Ti 4G独显)红,7099.0,【GTX1050Ti 4G独显】帧率高稳定性强运行更畅快,IPS防眩光显示屏全面还原游戏战场！,版本:游戏笔记本电脑,颜色:i5 8G GTX1050Ti PCIe 黑,颜色:i5 8G GTX1050Ti 白,颜色:i5 8G GTX1050Ti 高色域,颜色:i5 8G GTX1060 6G PCIe 黑,颜色:i5 8G GTX1060 6G 白,颜色:i5 8G GTX1060 6G 高色域,颜色:i5 8G GTX1060 6G 黑,颜色:i5-7300HQ 128G+1T GTX1050Ti 红,颜色:i7 16G GTX1060 白,颜色:i7 8G GTX1050Ti 白,颜色:i7 GTX1050Ti 高色域,颜色:i7 GTX1060 高色域,颜色:i9 16G GTX1060 白')
```

In [43]:

```
import os
import jieba
import jieba.posseg as pseg
import codecs

abspath = "/root/workspace/3.rs_project/project2/notebook"

stopwords_path = os.path.join(abspath, 'keywordExtract/extract/baidu_stopwords.txt')

# 结巴加载用户词典
userDict_path = os.path.join(abspath, "keywordExtract/extract/词典/all.txt")
jieba.load_userdict(userDict_path)

# 停用词文本
stopwords_path = os.path.join(abspath, "keywordExtract/extract/baidu_stopwords.txt")

def get_stopwords_list():
    """返回stopwords列表"""
    stopwords_list = [i.strip()
                      for i in codecs.open(stopwords_path).readlines()]
    return stopwords_list

# 所有的停用词列表
stopwords_list = get_stopwords_list()

class TextRank(jieba.analyse.TextRank):
    def __init__(self, window=20, word_min_len=2):#与某词关联的词小于2个，就不要了
        super(TextRank, self).__init__()
        self.span = window  # 窗口大小
        self.word_min_len = word_min_len  # 单词的最小长度
        # 要保留的词性，根据jieba github ，具体参见https://github.com/baidu/lac
        # set(可变集合)与frozenset(不可变集合)
        self.pos_filt = frozenset(
            ('n', 'x', 'eng', 'f', 's', 't', 'nr', 'ns', 'nt', "nw", "nz", "PER", "LOC

    def pairfilter(self, wp):
        """过滤条件，返回True或者False"""

        if wp.flag == "eng":
            if len(wp.word) <= 2:
                return False

        if wp.flag in self.pos_filt and len(wp.word.strip()) >= self.word_min_len \
                and wp.word.lower() not in stopwords_list:
            return True

text =  '''电脑,电脑整机,游戏本,戴尔DELL灵越游匣Master15.6英寸游戏笔记本电脑(i5-7300HQ

textrank_model = TextRank(window=10, word_min_len=2)
allowPOS = ('n', "x", 'eng', 'nr', 'ns', 'nt', "nw", "nz", "c")
tags = textrank_model.textrank(text, topK=20, withWeight=True, allowPOS=allowPOS, wi
print(tags)

```

```
[('颜色', 1.0), ('游戏', 0.712839905269392), ('GTX1050Ti', 0.6173173414936882), ('GTX
```

```
1060', 0.4941479481735383), ('笔记本电脑', 0.4619727692978662), ('电脑', 0.41432406308886194), ('高色域', 0.4106944624167243), ('戴尔', 0.371970464343393), ('DELL', 0.37081241281179783), ('游戏本', 0.339826427975927), ('游匣', 0.33646092621478985), ('Master15', 0.33535477423981386), ('英寸', 0.3021642579612178), ('战场', 0.28617607751165514), ('显示屏', 0.26646682099129176), ('版本', 0.2627075733325065), ('IPS', 0.2245590224235886), ('稳定性', 0.17327167036021401), ('PCIe', 0.1587201232207853), ('GSSD', 0.07646678807991644)]
```

运用

In [48]:

```
from functools import partial

def _mapPartitions(partition, industry):

    import os
    import jieba
    import jieba.posseg as pseg
    import codecs

    abspath = "/root/workspace/3.rs_project/project2/notebook"

    stopwords_path = os.path.join(abspath, 'keywordExtract/extract/baidu_stopwords.tx

    # 结巴加载用户词典
    userDict_path = os.path.join(abspath, "keywordExtract/extract/词典/all.txt")
    jieba.load_userdict(userDict_path)

    # 停用词文本
    stopwords_path = os.path.join(abspath, "keywordExtract/extract/baidu_stopwords.tx

    def get_stopwords_list():
        """返回stopwords列表"""
        stopwords_list = [i.strip()
                          for i in codecs.open(stopwords_path).readlines()]
        return stopwords_list

    # 所有的停用词列表
    stopwords_list = get_stopwords_list()

    class TextRank(jieba.analyse.TextRank):
        def __init__(self, window=20, word_min_len=2):
            super(TextRank, self).__init__()
            self.span = window  # 窗口大小
            self.word_min_len = word_min_len  # 单词的最小长度
            # 要保留的词性，根据jieba github ，具体参见https://github.com/baidu/lac
            self.pos_filt = frozenset(
                ('n', 'x', 'eng', 'f', 's', 't', 'nr', 'ns', 'nt', "nw", "nz", "PER",

        def pairfilter(self, wp):
            """过滤条件，返回True或者False"""

            if wp.flag == "eng":
                if len(wp.word) <= 2:
                    return False

            if wp.flag in self.pos_filt and len(wp.word.strip()) >= self.word_min_le
                    and wp.word.lower() not in stopwords_list:
                return True
    textrank_model = TextRank(window=10, word_min_len=2)
    allowPOS = ('n', "x", 'eng', 'nr', 'ns', 'nt', "nw", "nz", "c")

    for row in partition:
        tags = textrank_model.textrank(row.summary, topK=20, withWeight=True, allowPO
        for tag in tags:
            yield row.sku_id, industry, tag[0], tag[1]

```

```
57 mapPartitions = partial(_mapPartitions, industry="电子产品")
58
59 sku_tag_weights = ret.rdd.mapPartitions(mapPartitions)
60 sku_tag_weights = sku_tag_weights.toDF(["sku_id", "industry", "tag","weights"])
61 sku_tag_weights
```

Out[48]: DataFrame[sku_id: bigint, industry: string, tag: string, weights: double]

In [49]:
```
1 sku_tag_weights.show()
```

```
+------+--------+----+------------------+
|sku_id|industry| tag|           weights|
+------+--------+----+------------------+
|   148|  电子产品|  高度|               1.0|
|   148|  电子产品|  终端| 0.9883639010223446|
|   148|  电子产品|WPOS| 0.9468601431427544|
|   148|  电子产品| 触摸屏|  0.86760348122463|
|   148|  电子产品|  森锐| 0.8641422469338305|
|   148|  电子产品| 收银机| 0.8604138186822986|
|   148|  电子产品|  智能| 0.8531525111008649|
|   148|  电子产品|  业务| 0.8492750654830078|
|   148|  电子产品|  身份| 0.7319843682220414|
|   148|  电子产品|  包邮| 0.7023074720861229|
|   148|  电子产品| 读卡器| 0.6095231846952228|
|   148|  电子产品|  购物| 0.5376711388516783|
|   148|  电子产品|  正品| 0.5362617954388065|
|   148|  电子产品|数码配件| 0.4855585880270606|
|   148|  电子产品|  数码| 0.4839580467664243|
|   463|  电子产品| 读卡器|               1.0|
|   463|  电子产品|  颜色| 0.8520815042804352|
|   463|  电子产品|  安卓| 0.5855862956518749|
|   463|  电子产品|  手机|0.47418584773791483|
|   463|  电子产品|  电脑|0.43966846994965897|
+------+--------+----+------------------+
only showing top 20 rows
```

In [54]:
```
1 sku_tag_weights.where("tag='手机'").orderBy('weights').show()
```

```
+------+--------+---+--------------------+
|sku_id|industry|tag|             weights|
+------+--------+---+--------------------+
| 12084|  电子产品| 手机| 0.05433809439383953|
| 12000|  电子产品| 手机| 0.05503990756835343|
| 21242|  电子产品| 手机| 0.05928928206469161|
| 11987|  电子产品| 手机|0.059346698287006255|
| 11924|  电子产品| 手机| 0.06097968631781557|
| 11982|  电子产品| 手机| 0.06413152775972146|
| 12119|  电子产品| 手机| 0.06424278482093025|
| 11951|  电子产品| 手机| 0.06499619302992397|
| 12133|  电子产品| 手机|  0.0654244902308878|
| 12091|  电子产品| 手机| 0.06578279913350595|
| 11829|  电子产品| 手机| 0.06580457144700495|
| 12243|  电子产品| 手机| 0.06638587824849572|
| 11773|  电子产品| 手机| 0.06643819032693242|
| 20054|  电子产品| 手机| 0.06655695905971802|
| 19612|  电子产品| 手机| 0.06691817740931645|
| 19625|  电子产品| 手机| 0.06712183985354511|
| 11752|  电子产品| 手机| 0.06717762919850208|
| 24624|  电子产品| 手机| 0.06791438407558686|
| 20032|  电子产品| 手机| 0.06795459077587841|
| 20134|  电子产品| 手机| 0.06797542198694892|
+------+--------+---+--------------------+
only showing top 20 rows
```

2.2.4将提取的关键词存入hive

In [ ]:
```
1 sku_tag_wights.count()
```

In [56]:
```
1 sku_tag_weights.registerTempTable('tempTable')
2 spark.sql('desc tempTable').show()
```

```
+--------+---------+-------+
|col_name|data_type|comment|
+--------+---------+-------+
|  sku_id|   bigint|   null|
|industry|   string|   null|
|     tag|   string|   null|
| weights|   double|   null|
+--------+---------+-------+
```

In [57]:
```
# 创建hive表
sql = '''
create table if not exists sku_tag_weights(
sku_id int,
industry string,
tag string,
weights double
)
'''
spark.sql(sql)
```

Out[57]: DataFrame[]

In [58]:
```
spark.sql('insert into sku_tag_weights select * from tempTable')
```

Out[58]: DataFrame[]

In [59]:
```
spark.sql('select * from sku_tag_weights').show()
```

```
+------+--------+----+------------------+
|sku_id|industry| tag|           weights|
+------+--------+----+------------------+
|   148|  电子产品|  高度|               1.0|
|   148|  电子产品|  终端| 0.9883639010223446|
|   148|  电子产品|WPOS| 0.9468601431427544|
|   148|  电子产品| 触摸屏|  0.86760348122463|
|   148|  电子产品|  森锐| 0.8641422469338305|
|   148|  电子产品| 收银机| 0.8604138186822986|
|   148|  电子产品|  智能| 0.8531525111008649|
|   148|  电子产品|  业务| 0.8492750654830078|
|   148|  电子产品|  身份| 0.7319843682220414|
|   148|  电子产品|  包邮| 0.7023074720861229|
|   148|  电子产品| 读卡器| 0.6095231846952228|
|   148|  电子产品|  购物| 0.5376711388516783|
|   148|  电子产品|  正品| 0.5362617954388065|
|   148|  电子产品|数码配件| 0.4855585880270606|
|   148|  电子产品|  数码| 0.4839580467664243|
|   463|  电子产品| 读卡器|               1.0|
|   463|  电子产品|  颜色| 0.8520815042804352|
|   463|  电子产品|  安卓| 0.5855862956518749|
|   463|  电子产品|  手机|0.47418584773791483|
|   463|  电子产品|  电脑|0.43966846994965897|
+------+--------+----+------------------+
only showing top 20 rows
```