Paul 7 - enrich do…

```
%pyspark
```

FINISHED

```
# Zeppelin notebook to enrich domain summaries (from Paul 5) with examples (from Paul
    6)
# and topic metadata (from Tom 1)
# PJ - 9 November 2017

import boto
from pyspark.sql.types import *

# Load Domain Summaries DF in Gzip files (from Paul 5)
loadURL="s3://billsdata.net/CommonCrawl/domain_summaries6/" # Split into 20 files (as
    opposed to domain_summaries7, which is 10)
codec="org.apache.hadoop.io.compress.GzipCodec"
summary_df=spark.read.format('com.databricks.spark.csv').options(header='true', codec
    =codec).load(loadURL)
```

```
+--------------+--------+-------------+----------+-----------+--------------+------+----
--+
|payLevelDomain|numHosts|pldIsHostFlag|pldLinksIn|pldLinksOut|wasCrawledFlag|hcRank|prRa
nk|
+--------------+--------+-------------+----------+-----------+--------------+------+----
--+
|         37.ac|       1|         true|         1|       null|         false|49.503|47.7
23|
|      equal.ac|       1|         true|         3|          3|          true|69.571|68.0
98|
|      happy.ac|       1|         true|         2|          6|          true|19.030|69.9
99|
+--------------+--------+-------------+----------+-----------+--------------+------+----
--+
only showing top 3 rows
20
```

```
%pyspark
```

FINISHED

```
# Load examples dataframe (from Paul 6)
examplesURI="s3://billsdata.net/CommonCrawl/domain_examples4/"
example_df=spark.read.load(examplesURI)
example_df.show(10)
example_df.cache()
example_df.rdd.getNumPartitions()
```

```
+----+------------+-----------------+--------------------+
| PLD|exampleHosts|exampleInLinkPlds|  exampleOutLinkPlds|
+----+------------+-----------------+--------------------+
|null|        null|             null|     [instagram.com]|
|null|        null|             null|       [godaddy.com]|
|null|        null|             null|       [4.cn, 51.la]|
|null|        null|             null|[food.it, compro....|
|null|        null|             null|[web2printsoftwar...|
|null|        null|             null|[nu.nl, rtl.nl, g...|
|null|        null|             null|       [apdesign.be]|
|null|        null|             null|           [ovh.net]|
|null|        null|             null|[zonneschermenams...|
|null|        null|             null|    [googleapis.com]|
+----+------------+-----------------+--------------------+
only showing top 10 rows
204
```

```
%pyspark

# Remove square brackets from list output and remove Nulls
from pyspark.sql.functions import udf
def remove_brackets(egs):
    return str(egs).replace('[', '').replace(']', '')
print(remove_brackets("[bla, bla]"))
udf_remove_brackets = udf(remove_brackets, StringType())

example_df2=example_df.filter(example_df.PLD.isNotNull()).withColumn("tmp",
    udf_remove_brackets("exampleOutLinkPlds")).drop("exampleOutLinkPlds"
    ).withColumnRenamed("tmp","exampleOutLinkPlds")
example_df3=example_df2.withColumn("tmp", udf_remove_brackets("exampleInLinkPlds"
    )).drop("exampleInLinkPlds").withColumnRenamed("tmp","exampleInLinkPlds")
example_df4=example_df3.withColumn("tmp", udf_remove_brackets("exampleHosts")).drop
```

FINISHED

```
bla, bla
+------------------+--------------------+--------------------+------------------+
|               PLD|        exampleHosts|   exampleInLinkPlds|exampleOutLinkPlds|
+------------------+--------------------+--------------------+------------------+
|           0-0a.me|         dev.0-0a.me|     krijnhoetmer.nl|              None|
|    0-3-suikast.org|     0-3-suikast.org| list-of-domains.org|              None|
|         0-5-0.info|          0-5-0.info| list-of-domains.org|              None|
|            0-6.biz|             0-6.biz|     beyondwhois.com|              None|
|      0-6health.com|       0-6health.com|      allthecom.info|              None|
|            0-a.net|             0-a.net|      allthecom.info|              None|
|  0-apr-lifetime.cn|establish-credit-...|             fc2.com|              None|
|0-aprcreditcards.us| 0-aprcreditcards.us| list-of-domains.org|              None|
|     0-artlove.info|      0-artlove.info| list-of-domains.org|              None|
|         0-blog.com|     web2.0-blog.com|blogspot.com, seo...|              None|
+------------------+--------------------+--------------------+------------------+
only showing top 10 rows
90839924
```

```
%pyspark

# Join with Original summaries
example_df.unpersist()
example_summary_df=summary_df.join(example_df4, summary_df.payLevelDomain==example_df4
    .PLD).drop("PLD")
summary_df.unpersist()
example_summary_df.dropDuplicates().sort("numHosts",ascending=False).show(100)
```

FINISHED

```
|            c26b.com|     999|         true|        96|         35|          true|87.85
2|94.119|c26b.com, a1.c26b...|e26a.com, lineav....|cuiyu.info, macro...|
|          hkfr54.com|     999|         true|        15|         38|          true|89.65
1|77.550|hkfr54.com, a.hkf...|blogspot.com, klk...|yk272.com, kuku67...|
|            557i.com|     999|         true|       109|         64|          true|87.18
2|96.837|557i.com, x.557i....|5ccs.com, ek59.co...|8bss.com, ex57.co...|
|          jhfr67.com|     999|         true|        13|        455|          true|87.08
5|76.843|jhfr67.com, a.jhf...|kuku67.com, kkk26...|fwwd5.com, 38cc.c...|
|            557b.com|     999|         true|       108|         70|          true|93.51
2|96.842|557b.com, 110001....|5ccs.com, ek59.co...|557h.com, 9ttu.co...|
|              eur.vg|     999|        false|         3|       null|         false|97.25
8|64.636|a-flatwebdirector...|blogspot.com, blo...|                None|
|          hkfr55.com|     999|         true|        16|        268|          true|89.65
1|79.461|hkfr55.com, a.hkf...|blogspot.com, klk...|cilis.net, kk69mm...|
|          cbm665.com|     999|         true|        50|         19|          true|89.65
2|80.355|cbm665.com, a.cbm...|tyuqw59.com, bb19...|avmm088.com, hkk5...|
|           ywkqg.com|     999|         true|         2|         13|          true|17.17
3|63.071|ywkqg.com, 1136p....|8r7.info, ywxinyq...|youdao.com, igiyi...|
```

```
%pyspark

# Save Example Summaries as GZIP files, approx 100MB each.
#outURI="s3://billsdata.net/CommonCrawl/domain_summaries_withexamples4/"
#codec="org.apache.hadoop.io.compress.GzipCodec"
#example_summary_df.coalesce(40).write.format('com.databricks.spark.csv').options
    (header='true', codec=codec).save(outURI) # Creates 94x23.8MB files by default
```

FINISHED

```
NOT SAVING
```

```
%pyspark

# Load topic labels (from Tom 1)
loadURL="s3://billsdata.net/CommonCrawl/topic_model_1024_files
    /cc_index_page_topic_labels/" # TODO - switch to 2048!
topic_df=spark.read.load(loadURL)
topic_df.show(3)
print(topic_df.count())
topic_df.cache()
```

FINISHED

```
+--------------------+--------------------+--------------------+------------------+-----
--------------+--------------------+--------------------+--------------------+
|                host|                 url|              topic1|            score1|
topic2|             score2|              topic3|              score3|
+--------------------+--------------------+--------------------+------------------+-----
--------------+--------------------+--------------------+--------------------+
|billives.typepad.com|http://billives.t...|one_may_january_l...|0.3004067571438536|news_
business_dat...|0.17487022035440172|science_research_...| 0.12825866906681413|
|     docs.oracle.com|http://docs.oracl...|nbsp_windows_quot...|0.7843980004795321|forum
s_member_lik...|0.14079996570227268|add_cart_buy_pric...|0.058564617674095515|
|     e.il.tripod.com|http://e.il.tripo...|park_john_james_d...|0.7109430667614804|page_
collection_l...|0.08726735225206302|state_church_coll...|  0.0481260760704955|
+--------------------+--------------------+--------------------+------------------+-----
--------------+--------------------+--------------------+--------------------+
only showing top 3 rows
53526
171
```

```
%pyspark                                                                    FINISHED

# From Paul 5.

# Load in an uncompressed, partitioned format, for fast reading in the future
saveURI="s3://billsdata.net/CommonCrawl/hyperlinkgraph/cc-main-2017-may-jun-jul
    /domaingraph/vertices/"
#pld_df.coalesce(64).write.save(saveURI) # Use all default options
pld_df=spark.read.load(saveURI)
pld_df.show(3)
pld_df.cache()
```

```
+---+-------+
| ID|    PLD|
+---+-------+
|  0|  aaa.a|
|  1| aaa.aa|
|  2|aaa.aaa|
+---+-------+
only showing top 3 rows
DataFrame[ID: string, PLD: string]
```

```
%pyspark                                                                    FINISHED

# From Paul 5.

# Next, we'll construct a local dictionary from of all the PLDS (key is the PLD, value
    is the ID)
# This is our truth-table of known PLDs that we'll use when counting hosts
# Create a bloom filter using a pure python package (might be a little slow)
from pybloom import BloomFilter
pld_bf = BloomFilter(capacity=91000000, error_rate=0.005)
```

```
for row in pld_df.rdd.collect(): # limit(10000000) # TODO: Still bad (and exceeds
    spark.driver.maxResultSize with all rows)!
    pld_bf.add(row['PLD'])

print(pld_df.rdd.take(3))
print(pld_df.rdd.take(3)[2]['PLD'])
#pld_bf.add(pld_df.rdd.take(3)[2]['PLD'])
print("aaa.aaa" in pld_bf) # Should be True

import sys
print(sys.getsizeof(pld_bf))
print(len(pld_bf)) # Should match number of items entered

# Broadcast the bloom filter so it's available on all the slave nodes - we don't need
    to change
# it any more so it's fine being immutable.
pld_bf_distrib=sc.broadcast(pld_bf)

print("aaa.aaa" in pld_bf) # Should be true
```

```
[Row(ID=u'0', PLD=u'aaa.a'), Row(ID=u'1', PLD=u'aaa.aa'), Row(ID=u'2', PLD=u'aaa.aaa')]
aaa.aaa
True
64
90751305
True
False
True
False
```

```
%pyspark                                                                    FINISHED

# From Paul 5.

# Returns a Boolean to say whether PLD is a hostname in itself
def is_a_pld(hostname):
    #if hostname in pld_lookup_table:
    #if pld_lookup_table.filter(lambda a: a == hostname).count()>0:
    if hostname in pld_bf_distrib.value:
        return True
    else:
        return False

# Define a function to do the hostname->pld conversion, if the pld exists in our
    dictionary
def convert_hostname(hostname):
    # Return hostname as-is, if this is already a PLD
    #if hostname in pld_lookup_table:
    #if pld_lookup_table.filter(lambda a: a == hostname).count()>0:
    if hostname in pld_bf_distrib.value:
        return hostname
    # Otherwise we're going to have to split it up and test the parts
    try:
        parts=hostname.split('.')
        if (len(parts)>4 and is_a_pld('.'.join(parts[0:4]))):
```

```
            return '.'.join(parts[0:4])
        if (len(parts)>3 and is_a_pld('.'.join(parts[0:3]))):
            return '.'.join(parts[0:3])
        if (len(parts)>2 and is_a_pld('.'.join(parts[0:2]))):
            return '.'.join(parts[0:2])
        if (len(parts)>1 and is_a_pld('.'.join(parts[0:1]))):
            return '.'.join(parts[0:1])
        return "ERROR" # Couldn't find a corresponding PLD - this should never happen!
    except:
        return "ERROR"

udf_convert_hostname = udf(convert_hostname, StringType())

# Test
print(convert_hostname("aaa.aaa"))
```

```
aaa.aaa
True
```

```
%pyspark
```

FINISHED

```
# Function to reverse hostnames
from pyspark.sql.functions import udf
def reverse_domain(domain):
    return '.'.join(reversed(domain.split('.')))
print(reverse_domain("com.facebook"))
udf_reverse_domain = udf(reverse_domain, StringType())

# Convert hosts in Topic DF to PLDs using convert_hostname function from Paul 5.
topic_df2=topic_df.withColumn("pld",udf_reverse_domain(udf_convert_hostname
    (udf_reverse_domain("host")))) # Reverse the hostnames prior to lookup, then back
```

```
facebook.com
+--------------------+--------------------+--------------------+------------------+-----
---------------+--------------------+--------------------+--------------------+---------
------+
|                host|                 url|              topic1|            score1|
topic2|              score2|              topic3|              score3|            pld|
+--------------------+--------------------+--------------------+------------------+-----
---------------+--------------------+--------------------+--------------------+---------
------+
|billives.typepad.com|http://billives.t...|one_may_january_l...|0.30040675714385360|news_
business_dat...| 0.17487022035440172|science_research_...| 0.12825866906681413|    typep
ad.com|
|     docs.oracle.com|http://docs.oracl...|nbsp_windows_quot...|0.7843980004795321|forum
s_member_lik...| 0.14079996570227268|add_cart_buy_pric...|0.058564617674095515|     orac
le.com|
|     e.il.tripod.com|http://e.il.tripo...|park_john_james_d...|0.7109430667614804|page_
collection_l...| 0.08726735225206302|state_church_coll...|  0.0481260760704955|     trip
od.com|
```

```
%pyspark
```

FINISHED

```
# If multiple rows per PLD, only use the one with the shortest URL.
def url_len(url):
    return len(url)
print(url_len("com.facebook.bla"))
udf_url_len = udf(url_len, IntegerType())
from pyspark.sql.functions import col, desc
topic_df3=topic_df2.withColumn("url_len", udf_url_len("url"))
#topic_df3.show(30)

# Use window functions to filter to only the row with the shortest URL for each PLD
from pyspark.sql.window import Window
from pyspark.sql.functions import rank, col
window = Window.partitionBy(topic_df3['pld']).orderBy(topic_df3['url_len']) #
    Ascending order of URL length
topic_df4=topic_df3.select('*', rank().over(window).alias('rank')).filter(col('rank')
    <= 1)
topic_df4.show(20)
```

16

```
+--------------------+--------------------+--------------------+-------------------+----
---------------+-------------------+--------------------+--------------------+---------
----------+-------+----+
|                host|                 url|              topic1|             score1|
topic2|             score2|              topic3|              score3|                  pl
d|url_len|rank|
+--------------------+--------------------+--------------------+-------------------+----
---------------+-------------------+--------------------+--------------------+---------
----------+-------+----+
|www.aboutscotland...|http://www.abouts...|london_san_usa_fr...|0.38142474561098616|hote
ls_travel_hot...| 0.2536243335044581|hotel_hotels_room...| 0.12624115545257752|   abouts
cotland.com|     46|   1|
| www.azamgarh.nic.in|http://www.azamga...|law_state_court_t...|  0.609275047187203|home
_property_est...|0.19400757824203274|jobs_services_job...| 0.10178960136536272|     azam
garh.nic.in|     49|   1|
|www.chester-jense...|http://www.cheste...|united_magic_stat...| 0.5173311763910342|inc_
llc_company_n...|0.23671604167032806|post_reply_report...|0.003057378679439...|  chester
```

```
%pyspark                                                                RUNNING 0%

# Join on host/PLD
pld_df.unpersist()
topic_df2.unpersist()
topic_df3.unpersist()
enrich_summary_df=example_summary_df.join(topic_df4, example_summary_df.payLevelDomain
    ==topic_df.host)\
                                    .drop("host").drop("url").drop("score1").drop
                                        ("score2").drop("score3").drop("url_len").drop
                                        ("rank").drop("topic2").drop("topic3").drop
```

Started 3 minutes ago.

```
%pyspark                                                                FINISHED

# Save Example Summaries as GZIP files, approx 100MB each.
```

```
outURI="s3://billsdata.net/CommonCrawl/domain_summaries_withtopics1/"
codec="org.apache.hadoop.io.compress.GzipCodec"
enrich_summary_df.coalesce(1).write.format('com.databricks.spark.csv').options(header
```

```
%pyspark
```

READY