Projet Big Data Binomes:

- Wacim BELAHCEL, Formation Alternance
- Imad Oualid KACIMI, Formation Alternance

1. Mise en place de l'environnement de travail

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call dri
```

```
!pip install coclust
```

```
Requirement already satisfied: coclust in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (f
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: scipy in /usr/local/lib/python3.6/dist-packages (f
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.6/dist-pack
```

```
!pip install pandavro
```

```
Requirement already satisfied: pandavro in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: pandas>=1.1.5 in /usr/local/lib/python3.6/dist-pac
Requirement already satisfied: six>=1.9 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: numpy>=1.7.0 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: fastavro>=0.14.11 in /usr/local/lib/python3.6/dist
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.6
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-pack
```

```
!pip install pyarrow
```

```
Requirement already satisfied: pyarrow in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: numpy>=1.14 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: six>=1.0.0 in /usr/local/lib/python3.6/dist-packag
```

```
!pip install pyngrok
```

```
Requirement already satisfied: pyngrok in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: PyYAML in /usr/local/lib/python3.6/dist-packages (
```

```
!mkdir ~/.ngrok2
!echo "web_addr: localhost:5050" > ~/.ngrok2/ngrok.yml
```

```
mkdir: cannot create directory ‘/root/.ngrok2’: File exists
```

```
from pyngrok import ngrok

active_tunnels = ngrok.get_tunnels()
for tunnel in active_tunnels:
  public_url = tunnel.public_url
  ngrok.disconnect(public_url)

ngrok_tunnel = ngrok.connect(4040, return_ngrok_tunnel=True)
ngrok_tunnel
```

```
<NgrokTunnel: "http://8d5f2abc9222.ngrok.io" -> "http://localhost:4040">
```

```
# install Java8
!apt-get install openjdk-8-jdk-headless -qq > /dev/null
# download Spark
!wget -q https://downloads.apache.org/spark/spark-3.0.1/spark-3.0.1-bin-hadoop2.7.tgz
# unzip it
!tar xf spark-3.0.1-bin-hadoop2.7.tgz
# install findspark
!pip install -q findspark
```

```
# Set up required environment variables

import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-3.0.1-bin-hadoop2.7"
os.environ['PYSPARK_SUBMIT_ARGS'] = '--packages org.apache.spark:spark-avro_2.12:3.0.1
import findspark

findspark.init("spark-3.0.1-bin-hadoop2.7")
```

```
from pyspark import SparkContext, SparkConf
from pyspark.sql import SparkSession

conf = SparkConf().setAppName("mon application").setMaster("local[4]")
#spark context
sc = SparkContext(conf=conf)
spark = SparkSession.builder.config(conf=conf).getOrCreate()
```

2. Données :

2.a telechargement du jeu de données

```
! curl -L "http://qwone.com/~jason/20Newsgroups/20news-19997.tar.gz" > 20news-19997.ta
```

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 16.5M  100 16.5M    0     0  9617k      0  0:00:01  0:00:01 --:--:-- 9611k
```

2.b decompresser les données:

```
!tar xvzf 20news-19997.tar.gz
```

Le flux de sortie a été tronqué et ne contient que les 5000 dernières lignes.

```
20_newsgroups/soc.religion.christian/20607
20_newsgroups/soc.religion.christian/20608
20_newsgroups/soc.religion.christian/20609
20_newsgroups/soc.religion.christian/20579
20_newsgroups/soc.religion.christian/20580
20_newsgroups/soc.religion.christian/20581
20_newsgroups/soc.religion.christian/20582
20_newsgroups/soc.religion.christian/20583
20_newsgroups/soc.religion.christian/20584
20_newsgroups/soc.religion.christian/20585
20_newsgroups/soc.religion.christian/20586
20_newsgroups/soc.religion.christian/20587
20_newsgroups/soc.religion.christian/20588
20_newsgroups/soc.religion.christian/20589
20_newsgroups/soc.religion.christian/20590
20_newsgroups/soc.religion.christian/20591
20_newsgroups/soc.religion.christian/20592
20_newsgroups/soc.religion.christian/20593
20_newsgroups/soc.religion.christian/20594
20_newsgroups/soc.religion.christian/20595
20_newsgroups/soc.religion.christian/20596
20_newsgroups/soc.religion.christian/20597
20_newsgroups/soc.religion.christian/20598
20_newsgroups/soc.religion.christian/20599
20_newsgroups/soc.religion.christian/20600
20_newsgroups/soc.religion.christian/20601
20_newsgroups/soc.religion.christian/20602
20_newsgroups/soc.religion.christian/20603
20_newsgroups/soc.religion.christian/20604
20_newsgroups/soc.religion.christian/20605
20_newsgroups/soc.religion.christian/20610
20_newsgroups/soc.religion.christian/20611
20_newsgroups/soc.religion.christian/20612
20_newsgroups/soc.religion.christian/20613
20_newsgroups/soc.religion.christian/20614
20_newsgroups/soc.religion.christian/20615
20_newsgroups/soc.religion.christian/20616
20_newsgroups/soc.religion.christian/20617
20_newsgroups/soc.religion.christian/20618
20_newsgroups/soc.religion.christian/20619
20_newsgroups/soc.religion.christian/20620
20_newsgroups/soc.religion.christian/20621
20_newsgroups/soc.religion.christian/20622
20_newsgroups/soc.religion.christian/20623
20_newsgroups/soc.religion.christian/20624
20_newsgroups/soc.religion.christian/20625
20_newsgroups/soc.religion.christian/20626
20_newsgroups/soc.religion.christian/20627
```

```
20_newsgroups/soc.religion.christian/20628
20_newsgroups/soc.religion.christian/20629
20_newsgroups/soc.religion.christian/20630
20_newsgroups/soc.religion.christian/20631
20_newsgroups/soc.religion.christian/20632
20_newsgroups/soc.religion.christian/20633
20_newsgroups/soc.religion.christian/20634
20_newsgroups/soc.religion.christian/20635
20_newsgroups/soc.religion.christian/20636
20_newsgroups/soc.religion.christian/20637
```

2.c lecture des fichiers :

```
rdd_atheism = sc.wholeTextFiles("/content/20_newsgroups/alt.atheism/*")
rdd_baseball = sc.wholeTextFiles("/content/20_newsgroups/rec.sport.baseball/*")
```

2.d splite en entete et corp

```
import re
```

```
rdd_atheism_splitted = rdd_atheism.mapValues(lambda x:re.compile("Lines:\s*\d*\n").spl
rdd_baseball_splitted = rdd_baseball.mapValues(lambda x:re.compile("Lines:\s*\d*\n").s
```

```
rdd_atheism_splitted.take(1)
```

```
[('file:/content/20_newsgroups/alt.atheism/53517',
  ['Xref: cantaloupe.srv.cs.cmu.edu alt.atheism:53517 talk.religion.misc:83888 ta
   '\nIn article <C5L14I.JJ3@news.cso.uiuc.edu>, cobb@alexia.lis.uiuc.edu (Mike C
```

2.e extraire L'ensemble des entetes

Nous avons décidé d'extraire les entétes aprés avoir fusionner (suite plus bas).

```
def header_split(x,cat):

  if len(x) == 1:
    return None

  entete = x[0]
  past_key=""
  dict_entete = {}

  for line in entete.split("\n"):
    res = line.split(': ',1)
    if len(res) != 1:
      dict_entete[res[0]] = res[1]
```

```
        past_key = res[0]
      elif past_key != None :
        dict_entete[past_key] += res[0]

    dict_entete["corp_text"] = x[1]
    dict_entete["type_doc"] = cat
    return dict_entete


rdd_atheism_final =  rdd_atheism_splitted.mapValues(lambda x: header_split(x,"atheism"
rdd_baseball_final =  rdd_baseball_splitted.mapValues(lambda x: header_split(x,"baseba
```

```
rdd_atheism_final.count()
```

```
1000
```

```
rdd_baseball_final.count()
```

```
1000
```

2.f Fusion des rdd

```
rdd_final = rdd_atheism_final.union(rdd_baseball_final)
```

```
rdd_final.count()
```

```
2000
```

Rendu du rdd_final fusionné, avec l'ensemble des entetes (nous selectionnons nos entete par la suite), et le fichier concerné pour chaque individu<.

```
rdd_final.take(1)
```

```
[('file:/content/20_newsgroups/alt.atheism/53517',
  {'Date': 'Fri, 16 Apr 93 16:41:17 GMT',
   'From': 'eczcaw@mips.nott.ac.uk (C.Wainwright)',
   'Message-ID': '<1993Apr16.164117.6949@cs.nott.ac.uk>',
   'Newsgroups': 'alt.atheism,talk.religion.misc,talk.origins',
   'Organization': 'Nottingham University',
   'Path': 'cantaloupe.srv.cs.cmu.edu!das-news.harvard.edu!noc.near.net!news.cent
   'References': '<4fm9iYO00iV303voYt@andrew.cmu.edu> <C5Fuo2.FF8@news.cso.uiuc.e
   'Reply-To': 'eczcaw@mips.nott.ac.uk (C.Wainwright)',
   'Sender': 'news@cs.nott.ac.uk',
   'Subject': 'Re: After 2000 years, can we say that Christian Morality is ',
   'Xref': 'cantaloupe.srv.cs.cmu.edu alt.atheism:53517 talk.religion.misc:83888
   'corp_text': '\nIn article <C5L14I.JJ3@news.cso.uiuc.edu>, cobb@alexia.lis.uiu
   'type_doc': 'atheism'})]
```

Suppressions des individu ne contenant aucun entete.

```
rdd_final = rdd_final.filter(lambda x: x[1] != None)
```

2.e Extractions des entetes que nous souhaitons conserver.

fonctions utilitaire pour choix des entetes à conserver.

```
def Union(lst1, lst2):
    final_list = list(set(lst1) | set(lst2))
    return final_list


def intersection(lst1, lst2):
    lst3 = [value for value in lst1 if value in lst2]
    return lst3

def Intersection(lst1, lst2):
    return set(lst1).intersection(lst2)
```

extraction des clés des entétes.

```
rdd_keys = rdd_final.map(lambda x: list(x[1].keys()))
```

Extraction de l'ensemble des Entétes existantes.

```
shared_keys_union = rdd_keys.reduce(lambda x,y : Union(x,y))
shared_keys_union
```

```
['X-Disclaimer',
 'Followup-To',
 'Article-I.D.',
 'NNTP-Posting-Host',
 'Nntp-Posting-User',
 ' References',
 'X-News-Reader',
 'X-XXDate',
 'References',
 'Organization',
 'Nntp-Posting-Host',
 'Message-ID',
 'X-Mailer',
 'To',
 'type_doc',
 'X-XXMessage-ID',
 'News-Software',
 'Summary',
 'Xref',
 'Sender',
```

```
      'Expires',
      'Distribution',
      'Originator',
      'Subject',
      'In-Reply-To',
      'corp_text',
      'X-Newsreader',
      'Supersedes',
      'Date',
      'From',
      'Disclaimer',
      'Approved',
      'In-reply-to',
      'X-UserAgent',
      'X-Sender',
      'Reply-To',
      'Path',
      'Nntp-Posting-Host-[nntpd-681]',
      'X-Posted-From',
      'Newsgroups',
      'Keywords']
```

Extractions des intersections (Entétes qui se trouvent dans l'ensemble des fichiers), nous avons decidé de conserver les intersection et non pas les unions en ajoutant "Organization, summary, keywords" que nous estimons étre des entétes interessantes.

```
shared_keys_intersection = rdd_keys.reduce(lambda x,y : Intersection(x,y))
shared_keys_intersection = list(shared_keys_intersection)
shared_keys_intersection += ["Organization", "Summary","Keywords"]
```

```
shared_keys_intersection
```

```
     ['Subject',
      'Path',
      'corp_text',
      'From',
      'Newsgroups',
      'type_doc',
      'Organization',
      'Summary',
      'Keywords']
```

```
import numpy as np
import os
#ajouter les clés manquante aux individu et les set à None
def add_keys_header(x,shared_keys):

  keys_to_add = np.setdiff1d(shared_keys,list(x[1].keys()))

  for keys in keys_to_add:
    x[1][keys] = None
  x[1]["id"] =os.path.basename(x[0])
  return x[1]

#ne garder que les clés qui se trouves dans "shared_keys"
```

```
def remove_unshared_keys(x,shared_keys):

  d = {}

  for keys in shared_keys:
    d[keys]= x[keys]
  d["id"]=  x["id"]
  return d
```

Selection des entétes

```
rdd_final = rdd_final.map(lambda x: add_keys_header(x,shared_keys_intersection)).map(l
```

```
rdd_final.take(3)
```

```
[{'From': 'eczcaw@mips.nott.ac.uk (C.Wainwright)',
  'Keywords': None,
  'Newsgroups': 'alt.atheism,talk.religion.misc,talk.origins',
  'Organization': 'Nottingham University',
  'Path': 'cantaloupe.srv.cs.cmu.edu!das-news.harvard.edu!noc.near.net!news.cente
  'Subject': 'Re: After 2000 years, can we say that Christian Morality is ',
  'Summary': None,
  'corp_text': '\nIn article <C5L14I.JJ3@news.cso.uiuc.edu>, cobb@alexia.lis.uiuc
  'id': '53517',
  'type_doc': 'atheism'},
 {'From': 'kmr4@po.CWRU.edu (Keith M. Ryan)',
  'Keywords': None,
  'Newsgroups': 'alt.atheism',
  'Organization': 'Case Western Reserve University',
  'Path': 'cantaloupe.srv.cs.cmu.edu!crabapple.srv.cs.cmu.edu!bb3.andrew.cmu.edu!
  'Subject': 'Re: free moral agency',
  'Summary': None,
  'corp_text': 'Distribution: na\nMessage-ID: <kmr4.1679.735522637@po.CWRU.edu>\r
  'id': '54221',
  'type_doc': 'atheism'},
 {'From': 'kmr4@po.CWRU.edu (Keith M. Ryan)',
  'Keywords': None,
  'Newsgroups': 'talk.abortion,alt.atheism,talk.religion.misc',
  'Organization': 'Case Western Reserve University',
  'Path': 'cantaloupe.srv.cs.cmu.edu!das-news.harvard.edu!noc.near.net!howland.re
  'Subject': 'Re: After 2000 years, can we say that Christian Morality is',
  'Summary': None,
  'corp_text': 'Message-ID: <kmr4.1587.734911207@po.CWRU.edu>\nReferences: <1993A
  'id': '53089',
  'type_doc': 'atheism'}]
```

2.g transformer le rdd pour que chaque element soit de type pyspark.sql.Row

```
from pyspark.sql import Row
df = rdd_final.map(lambda x: Row(**x)).toDF()
```

2.h créer un objet de type dataframe à partir du rdd

```
df = rdd_final.map(lambda x: Row(**x)).toDF()
```

```
df.show()
```

```
+--------------------+--------------------+--------------------+----------------
|             Subject|                Path|           corp_text|                F
+--------------------+--------------------+--------------------+----------------
|Re: After 2000 ye...|cantaloupe.srv.cs...|
In article <C5L1...|eczcaw@mips.nott....|alt.atheism,talk....| atheism|Nottingham
|Re: free moral ag...|cantaloupe.srv.cs...|Distribution: na
...|kmr4@po.CWRU.edu ...|         alt.atheism| atheism|Case Western Rese...|   nu
|Re: After 2000 ye...|cantaloupe.srv.cs...|Message-ID: <kmr4...|kmr4@po.CWRU.edu
|Re: Amusing athei...|cantaloupe.srv.cs...|Distribution: wor...|cfaehl@vesta.unm.
|Re: Yet more Rush...|cantaloupe.srv.cs...|
In article <1993...|jaeger@buphy.bu.e...|         alt.atheism| atheism|Boston Uni
|Re: The Inimitabl...|cantaloupe.srv.cs...|Distribution: wor...|kmr4@po.CWRU.edu
|Re: Christian Mor...|cantaloupe.srv.cs...|
In article <pww-...|sandvik@newton.ap...|         alt.atheism| atheism|Cookamunga
|   Re: <<Pompous ass|cantaloupe.srv.cs...|
In article <1ql6...|bobbe@vice.ICO.TE...|         alt.atheism| atheism|Tektronix
|Re: After 2000 ye...|cantaloupe.srv.cs...|
In <1993Apr15.07...|cobb@alexia.lis.u...|alt.atheism,talk....| atheism|University
|Re: Christian Mor...|cantaloupe.srv.cs...|
In article <1993...|acooper@mac.cc.ma...|         alt.atheism| atheism|  Macalest
|[UPI] "Mother fil...|cantaloupe.srv.cs...|
[By default, fol...|kadie@cs.uiuc.edu...|rec.scouting,soc....| atheism|University
|Re: Theism and Fa...|cantaloupe.srv.cs...|Message-ID: <1r0s...|frank@D012S658.uu
|Re: free moral ag...|cantaloupe.srv.cs...|Message-ID: <kmr4...|kmr4@po.CWRU.edu
|  Re: some thoughts.|cantaloupe.srv.cs...|Message-ID: <1qla...|keith@cco.caltech
|Re: Omnipotence (...|cantaloupe.srv.cs...|Message-ID: <kmr4...|kmr4@po.CWRU.edu
|Re: Is Keith as i...|cantaloupe.srv.cs...|Message-ID: <1pm6...|keith@cco.caltech
|Re: Theism and Fa...|cantaloupe.srv.cs...|
In article <1r0s...|I3150101@dbstu1.r...|         alt.atheism| atheism|Technical
|Re: thoughts on c...|cantaloupe.srv.cs...|
Ed McCreary (edm...|bil@okcforum.osrh...|         alt.atheism| atheism|Okcforum U
|Re: thoughts on c...|cantaloupe.srv.cs...|
cmtan@iss.nus.sg...|dfuller@portal.hq...|         alt.atheism| atheism|      Vide
|Re: After 2000 ye...|cantaloupe.srv.cs...|Message-ID: <1qkn...|frank@D012S658.uu
+--------------------+--------------------+--------------------+----------------
only showing top 20 rows
```

save in avro parquet :

2.i Avro

```
import pandavro as pdx
import pandas as pd
import numpy as np

avro_filename = "df.avro"
pdx.to_avro(avro_filename, df.toPandas())
```

```
saved = pdx.read_avro(avro_filename)
print(saved)
```

```
                                                Subject  ...      id
0     Re: After 2000 years, can we say that Christia...  ...   53517
1                                Re: free moral agency  ...   54221
2     Re: After 2000 years, can we say that Christia...  ...   53089
3                    Re: Amusing atheists and agnostics  ...   53257
4                Re: Yet more Rushdie [Re: ISLAMIC LAW]  ...   53758
...                                                 ...  ...     ...
1989                                           Moe Berg  ...  104699
1990                        PHILS, NL EAST NOT SO WEAK  ...  104570
1991  Re: Strike zone width (was Re: Jose Canseco's ...  ...  105299
1992                          Re: Early BBDDD Returns?  ...  104449
1993                                       Re: Mel Hall  ...  104424

[1994 rows x 10 columns]
```

2.j Parquet

```
import pyarrow as pa
import pyarrow.parquet as pq

parquet_filename = "df.parquet"
table = pa.Table.from_pandas(df.toPandas())
pq.write_table(table, parquet_filename)
```

```
table2 = pq.read_table(parquet_filename)
table2.to_pandas()
```

	Subject	Path	
0	Re: After 2000 years, can we say that Christia...	cantaloupe.srv.cs.cmu.edu!das-news.harvard.edu...	<C5L14I.JJ3@news.cs
1	Re: free moral agency	cantaloupe.srv.cs.cmu.edu!crabapple.srv.cs.cmu...	Distribution: n <kmr
2	Re: After 2000 years, can we say that Christia...	cantaloupe.srv.cs.cmu.edu!das-news.harvard.edu...	<kmr4.1587.734911207@pc
3	Re: Amusing atheists and agnostics	cantaloupe.srv.cs.cmu.edu!magnesium.club.cc.cm...	Distribution: wor <
4	Re: Yet more Rushdie [Re: ISLAMIC LAW]	cantaloupe.srv.cs.cmu.edu!das-news.harvard.edu...	<1993Apr15.215833.159
...	...	...	
1989	Moe Berg	cantaloupe.srv.cs.cmu.edu!magnesium.club.cc.cm...	\nNPR this morning had a
1990	PHILS, NL EAST NOT SO WEAK	cantaloupe.srv.cs.cmu.edu!rochester!udel!gatec...	\nI Love it how all of t
1991	Re: Strike zone width (was Re: Jose Canseco's	cantaloupe.srv.cs.cmu.edu!magnesium.club.cc.cm...	<2685.2bd51686@atlas

comme on peut le voir, le fichier parquet est 4 fois moins lourd que le fichier avro.

Re: Early

```
!ls -lh
```

```
total 233M
-rw-r--r--  1 root  root   17M Feb 19 22:06 20news-19997.tar.gz
drwxr-xr-x 22 28757 staff 4.0K Apr  3  1999 20_newsgroups
-rw-r--r--  1 root  root  3.8M Feb 19 22:33 df.avro
-rw-r--r--  1 root  root  2.2M Feb 19 22:33 df.parquet
```

```
drwx------  5 root  root  4.0K Feb 19 19:52 drive
drwxr-xr-x  1 root  root  4.0K Feb 16 16:35 sample_data
-rw-r--r--  1 root  root   120 Feb 19 21:28 sample_kmeans_data.txt
-rw-r--r--  1 root  root  103K Feb 19 21:52 sample_libsvm_data.txt
drwxr-xr-x 13  1000  1000 4.0K Aug 28 08:10 spark-3.0.1-bin-hadoop2.7
-rw-r--r--  1 root  root  210M Aug 28 09:25 spark-3.0.1-bin-hadoop2.7.tgz
drwxr-xr-x  3 root  root  4.0K Feb 19 21:33 target
```

3. Analyse descriptive:

```
df.select("type_doc","Newsgroups","id").show()
```

```
+--------+--------------------+-----+
|type_doc|          Newsgroups|   id|
+--------+--------------------+-----+
| atheism|alt.atheism,talk....|53517|
| atheism|         alt.atheism|54221|
| atheism|talk.abortion,alt...|53089|
| atheism|         alt.atheism|53257|
| atheism|         alt.atheism|53758|
| atheism|         alt.atheism|54227|
| atheism|         alt.atheism|53473|
| atheism|         alt.atheism|53230|
| atheism|alt.atheism,talk....|53148|
| atheism|         alt.atheism|53537|
| atheism|rec.scouting,soc....|54234|
| atheism|         alt.atheism|53293|
| atheism|         alt.atheism|54219|
| atheism|         alt.atheism|53201|
| atheism|         alt.atheism|51197|
| atheism|         alt.atheism|51232|
| atheism|         alt.atheism|53594|
| atheism|         alt.atheism|53438|
| atheism|         alt.atheism|53158|
| atheism|talk.abortion,alt...|53091|
+--------+--------------------+-----+
only showing top 20 rows
```

3.a verifier qu'on a bien 2 catégories

```
from pyspark.sql.functions import col, countDistinct
df.groupBy("type_doc").count().show()
df.groupBy("Organization").count().sort(col("count").desc()).show()
```

```
+--------+-----+
|type_doc|count|
+--------+-----+
|baseball|  994|
| atheism| 1000|
+--------+-----+
```

```
+--------------------+-----+
|        Organization|count|
+--------------------+-----+
|                null|  112|
|                 sgi|   70|
|California Instit...|   65|
|  Siemens-Nixdorf AG|   41|
|Mantis Consultant...|   40|
|Case Western Rese...|   39|
|Technical Univers...|   35|
|Okcforum Unix Use...|   32|
|University of Ill...|   32|
|Cookamunga Touris...|   25|
|Cornell Univ. CS ...|   25|
|Princeton University|   24|
|Netcom Online Com...|   23|
|Boston University...|   21|
|Tektronix Inc., B...|   20|
|   Allegheny College|   19|
|University of Ill...|   19|
|University of Den...|   18|
|University of Not...|   17|
|  Indiana University|   17|
+--------------------+-----+
only showing top 20 rows
```

3.b donner le nombre d'organisations differentes

```
df.select(countDistinct("Organization")).show()
```

```
+----------------------------+
|count(DISTINCT Organization)|
+----------------------------+
|                         467|
+----------------------------+
```

3.c Autre analyses descriptive

nombre d'element null par entete

```
from pyspark.sql.functions import isnan, when, count, col

df.select([count(when(isnan(c) | col(c).isNull(), c)).alias(c) for c in df.columns]).s
```

```
+-------+----+---------+----+----------+--------+------------+-------+--------+--
|Subject|Path|corp_text|From|Newsgroups|type_doc|Organization|Summary|Keywords| i
+-------+----+---------+----+----------+--------+------------+-------+--------+--
|      0|   0|        0|   0|         0|       0|         112|   1960|    1937|
+-------+----+---------+----+----------+--------+------------+-------+--------+--
```

Fill les elements null de organisation(pour qu'ils ne soient plus considéré comme null)

```
df = df.fillna( { 'Organization':""} )
```

```
df.select([count(when(isnan(c) | col(c).isNull(), c)).alias(c) for c in df.columns]).s
```

```
+-------+----+---------+----+----------+--------+------------+-------+--------+--
|Subject|Path|corp_text|From|Newsgroups|type_doc|Organization|Summary|Keywords| i
+-------+----+---------+----+----------+--------+------------+-------+--------+--
|      0|   0|        0|   0|         0|       0|           0|   1960|    1937|
+-------+----+---------+----+----------+--------+------------+-------+--------+--
```

compter l'occurence de chaque sujet et ordonner par ordre décroissant

```
df.groupBy("Subject").count().sort(col("count").desc()).show()
```

```
+--------------------+-----+
|             Subject|count|
+--------------------+-----+
|Re: After 2000 ye...|  130|
|Re: <Political At...|   46|
|Re: Yet more Rush...|   36|
|     Re: Jack Morris|   34|
|  Re: some thoughts.|   31|
|Re: Jewish Baseba...|   30|
|Re: free moral ag...|   27|
|Re: Rawlins debun...|   27|
|Re: Amusing athei...|   22|
|Re: Christian Mor...|   20|
|Re: Genocide is C...|   18|
|    Re: Albert Sabin|   18|
|Re: The Inimitabl...|   16|
|  Re: Young Catchers|   16|
|Re: YOU WILL ALL ...|   15|
|Re: HBP? BB? BIG-...|   14|
|Re: islamic autho...|   14|
|Re: And America's...|   14|
|Re: Notes on Jays...|   14|
|Re: Braves Pitchi...|   13|
+--------------------+-----+
only showing top 20 rows
```

4 / 5 / 6 Transformation du texte et clustering

Suppressions de colonne inutile lors du clustering (trop de null ou hors du sujet)

```
from pyspark.sql.functions import concat, col, lit
```

```
columns_to_drop = ["Path","From","Summary","Keywords","id"]
df = df.drop(*columns_to_drop)
```

```
df.show()
```

```
+--------------------+--------------------+--------------------+--------+--------
|             Subject|           corp_text|          Newsgroups|type_doc|
+--------------------+--------------------+--------------------+--------+--------
|Re: After 2000 ye...|
In article <C5L1...|alt.atheism,talk....| atheism|Nottingham Univer...|
|Re: free moral ag...|Distribution: na
...|         alt.atheism| atheism|Case Western Rese...|
|Re: After 2000 ye...|Message-ID: <kmr4...|talk.abortion,alt...| atheism|Case Wes
|Re: Amusing athei...|Distribution: wor...|         alt.atheism| atheism|Universi
|Re: Yet more Rush...|
In article <1993...|         alt.atheism| atheism|Boston University...|
|Re: The Inimitabl...|Distribution: wor...|         alt.atheism| atheism|Case Wes
|Re: Christian Mor...|
In article <pww-...|         alt.atheism| atheism|Cookamunga Touris...|
|   Re: <<Pompous ass|
In article <1ql6...|         alt.atheism| atheism|Tektronix Inc., B...|
|Re: After 2000 ye...|
In <1993Apr15.07...|alt.atheism,talk....| atheism|University of Ill...|
|Re: Christian Mor...|
In article <1993...|         alt.atheism| atheism|  Macalester College|
|[UPI] "Mother fil...|
[By default, fol...|rec.scouting,soc....| atheism|University of Ill...|
|Re: Theism and Fa...|Message-ID: <1r0s...|         alt.atheism| atheism|  Siemer
|Re: free moral ag...|Message-ID: <kmr4...|         alt.atheism| atheism|Case Wes
|  Re: some thoughts.|Message-ID: <1qla...|         alt.atheism| atheism|Californ
|Re: Omnipotence (...|Message-ID: <kmr4...|         alt.atheism| atheism|Case Wes
|Re: Is Keith as i...|Message-ID: <1pm6...|         alt.atheism| atheism|Californ
|Re: Theism and Fa...|
In article <1r0s...|         alt.atheism| atheism|Technical Univers...|
|Re: thoughts on c...|
Ed McCreary (edm...|         alt.atheism| atheism|Okcforum Unix Use...|
|Re: thoughts on c...|
cmtan@iss.nus.sg...|         alt.atheism| atheism|      VideOcart Inc.|
|Re: After 2000 ye...|Message-ID: <1qkn...|talk.abortion,alt...| atheism|  Siemer
+--------------------+--------------------+--------------------+--------+--------
only showing top 20 rows
```

transformation des caractéres en miniscule

```
from pyspark.sql.functions import lower, col
```

```
df_final = df.select(lower(concat(col("corp_text"),lit('\n'),col("Subject"),lit('\n'),

df_final.show()
```

```
+--------------------+
|            FullText|
+--------------------+
|
in article <c5l1...|
|distribution: na
...|
|message-id: <kmr4...|
|distribution: wor...|
|
in article <1993...|
|distribution: wor...|
|
in article <pww-...|
|
in article <1ql6...|
|
in <1993apr15.07...|
|
in article <1993...|
|
[by default, fol...|
|message-id: <1r0s...|
|message-id: <kmr4...|
|message-id: <1qla...|
|message-id: <kmr4...|
|message-id: <1pm6...|
|
in article <1r0s...|
|
ed mccreary (edm...|
|
cmtan@iss.nus.sg...|
|message-id: <1qkn...|
+--------------------+
only showing top 20 rows
```

4.a + b + autres traitements:

- découper les documents en liste à l'aide du tokenizer
- suppression de stopords
- créer une representation vectoriel à l'aide de hashing TF
- regularisation L2norm

```
from pyspark.ml.feature import HashingTF, IDF, Tokenizer
from pyspark.ml.feature import RegexTokenizer, StopWordsRemover, HashingTF, IDF, Norma

tokenizer = Tokenizer(inputCol="FullText", outputCol="words")
wordsData = tokenizer.transform(df_final)
```

```
from pyspark.ml.feature import StopWordsRemover
remover = StopWordsRemover(inputCol="words", outputCol="filtered")
removedwordsData  = remover.transform(wordsData)
removedwordsData .select("filtered").show(truncate=False)
```

```
+--------------------------------------------------------------------------------
|filtered
+--------------------------------------------------------------------------------
|[, article, <c5l14i.jj3@news.cso.uiuc.edu>,, cobb@alexia.lis.uiuc.edu, (mike, cc
|[distribution:, na, message-id:, <kmr4.1679.735522637@po.cwru.edu>, references:,
|[message-id:, <kmr4.1587.734911207@po.cwru.edu>, references:, <1993apr15.071814.
|[distribution:, world, message-id:, <1r10jcinnt1g@lynx.unm.edu>, references:, <t
|[, article, <1993apr15.215833.15970@bnr.ca>, (rashid), writes:, , , >>, twelve,
|[distribution:, world,public, message-id:, <kmr4.1620.735098175@po.cwru.edu>, re
|[, article, <pww-210493010443@spac-at1-59.rice.edu>,, pww@spacsun.rice.edu, (pet
|[, article, <1ql6jiinn5df@gap.caltech.edu>, keith@cco.caltech.edu, (keith, allar
|[, <1993apr15.074615.957@abo.fi>, mandtbacka@finabo.abo.fi, (mats, andtbacka), ,
|[, article, <1993apr21.140649.5660@cs.nott.ac.uk>,, kax@cs.nott.ac.uk, (kevin, a
|[, [by, default,, followups, 3, newsgroups.], , short, excerpt:, , >, brookfielc
|[message-id:, <1r0sn0$3r@horus.ap.mchp.sni.de>, references:, <16bb511ba2.i315010
|[message-id:, <kmr4.1677.735522451@po.cwru.edu>, references:, <1qugbk$em7@groupe
|[message-id:, <1qla3dinn6p0@gap.caltech.edu>, references:, <bissda.4.734849678@s
|[message-id:, <kmr4.1443.734058355@po.cwru.edu>, references:, <2942949719.2.p002
|[message-id:, <1pm6gpinnm4v@gap.caltech.edu>, references:, <1pa7aeinnsa9@gap.cal
|[, article, <1r0sn0$3r@horus.ap.mchp.sni.de>, frank@d012s658.uucp, (frank, o'dwy
|[, ed, mccreary, (edm@twisto.compaq.com), wrote:, :, >>>>>, 16, apr, 93, 05:10:1
|[, cmtan@iss.nus.sg, (tan, chade, meng, -, dan), writes:, :, , [, ., ., ., ., .,
|[message-id:, <1qkn1t$59l@horus.ap.mchp.sni.de>, references:, <sandvik-140493230
+--------------------------------------------------------------------------------
only showing top 20 rows
```

haghing TF

```
hashingTF = HashingTF(inputCol="filtered", outputCol="rawFeatures", numFeatures=1000)
featurizedData  = hashingTF.transform(removedwordsData)
```

```
idf = IDF(inputCol="rawFeatures", outputCol="features")
idfModel = idf.fit(featurizedData)
rescaledData  = idfModel.transform(featurizedData )
```

```
normalizer = Normalizer(inputCol="features", outputCol="normFeatures")
l2NormData = normalizer.transform(rescaledData)
```

Resultat final apres traitement

```
l2NormData.show()
```

```
+--------------------+--------------------+--------------------+----------------
|            FullText|               words|            filtered|         rawFeatu
```

```
+--------------------+--------------------+--------------------+----------------
|
in article <c5l1...|[, in, article, <...|[, article, <c5l1...|(1000,[3,7,27,39,..
|distribution: na
...|[distribution:, n...|[distribution:, n...|(1000,[2,39,47,73...|(1000,[2,39,47
|message-id: <kmr4...|[message-id:, <km...|[message-id:, <km...|(1000,[4,18,20,24
|distribution: wor...|[distribution:, w...|[distribution:, w...|(1000,[1,8,12,15,
|
in article <1993...|[, in, article, <...|[, article, <1993...|(1000,[10,12,13,1..
|distribution: wor...|[distribution:, w...|[distribution:, w...|(1000,[6,22,26,61
|
in article <pww-...|[, in, article, <...|[, article, <pww-...|(1000,[4,11,20,23..
|
in article <1ql6...|[, in, article, <...|[, article, <1ql6...|(1000,[12,15,20,3..
|
in <1993apr15.07...|[, in, <1993apr15...|[, <1993apr15.074...|(1000,[3,7,9,10,1..
|
in article <1993...|[, in, article, <...|[, article, <1993...|(1000,[4,7,19,20,..
|
[by default, fol...|[, [by, default,,...|[, [by, default,,...|(1000,[19,20,40,5..
|message-id: <1r0s...|[message-id:, <1r...|[message-id:, <1r...|(1000,[3,4,7,11,1
|message-id: <kmr4...|[message-id:, <km...|[message-id:, <km...|(1000,[6,7,33,34,
|message-id: <1qla...|[message-id:, <1q...|[message-id:, <1q...|(1000,[0,12,25,29
|message-id: <kmr4...|[message-id:, <km...|[message-id:, <km...|(1000,[10,27,38,5
|message-id: <1pm6...|[message-id:, <1p...|[message-id:, <1p...|(1000,[24,49,78,8
|
in article <1r0s...|[, in, article, <...|[, article, <1r0s...|(1000,[0,2,3,4,7,..
|
ed mccreary (edm...|[, ed, mccreary, ...|[, ed, mccreary, ...|(1000,[7,20,25,32..
|
cmtan@iss.nus.sg...|[, cmtan@iss.nus....|[, cmtan@iss.nus....|(1000,[10,11,12,2..
|message-id: <1qkn...|[message-id:, <1q...|[message-id:, <1q...|(1000,[2,4,17,23,
+--------------------+--------------------+--------------------+----------------
only showing top 20 rows
```

5 utilier l'algorithme kmean de spark

```
%%time
from pyspark.ml.clustering import KMeans
from pyspark.ml.evaluation import ClusteringEvaluator
import pandas as pd
```

```
kmeans = KMeans(featuresCol="normFeatures").setK(2).setSeed(1)
model = kmeans.fit(l2NormData)
```

```
CPU times: user 45.9 ms, sys: 7.27 ms, total: 53.2 ms
Wall time: 7.57 s
```

```
# Make predictions
predictions = model.transform(l2NormData)
```

```
predictions = model.transform(l2NormData)

# Evaluate clustering by computing Silhouette score
evaluator = ClusteringEvaluator()

silhouette = evaluator.evaluate(predictions)
print("Silhouette with squared euclidean distance = " + str(silhouette))

centers = model.clusterCenters()
```

```
Silhouette with squared euclidean distance = -0.2795142616658651
```

```
predictions.show()
```

```
+--------------------+--------------------+--------------------+----------------
|            FullText|               words|            filtered|         rawFeatu
+--------------------+--------------------+--------------------+----------------
|
in article <c5l1...|[, in, article, <...|[, article, <c5l1...|(1000,[3,7,27,39,..
|distribution: na
...|[distribution:, n...|[distribution:, n...|(1000,[2,39,47,73...|(1000,[2,39,47
|message-id: <kmr4...|[message-id:, <km...|[message-id:, <km...|(1000,[4,18,20,24
|distribution: wor...|[distribution:, w...|[distribution:, w...|(1000,[1,8,12,15,
|
in article <1993...|[, in, article, <...|[, article, <1993...|(1000,[10,12,13,1..
|distribution: wor...|[distribution:, w...|[distribution:, w...|(1000,[6,22,26,61
|
in article <pww-...|[, in, article, <...|[, article, <pww-...|(1000,[4,11,20,23..
|
in article <1ql6...|[, in, article, <...|[, article, <1ql6...|(1000,[12,15,20,3..
|
in <1993apr15.07...|[, in, <1993apr15...|[, <1993apr15.074...|(1000,[3,7,9,10,1..
|
in article <1993...|[, in, article, <...|[, article, <1993...|(1000,[4,7,19,20,..
|
[by default, fol...|[, [by, default,,...|[, [by, default,,...|(1000,[19,20,40,5..
|message-id: <1r0s...|[message-id:, <1r...|[message-id:, <1r...|(1000,[3,4,7,11,1
|message-id: <kmr4...|[message-id:, <km...|[message-id:, <km...|(1000,[6,7,33,34,
|message-id: <1qla...|[message-id:, <1q...|[message-id:, <1q...|(1000,[0,12,25,29
|message-id: <kmr4...|[message-id:, <km...|[message-id:, <km...|(1000,[10,27,38,5
|message-id: <1pm6...|[message-id:, <1p...|[message-id:, <1p...|(1000,[24,49,78,8
|
in article <1r0s...|[, in, article, <...|[, article, <1r0s...|(1000,[0,2,3,4,7,..
|
ed mccreary (edm...|[, ed, mccreary, ...|[, ed, mccreary, ...|(1000,[7,20,25,32..
|
cmtan@iss.nus.sg...|[, cmtan@iss.nus....|[, cmtan@iss.nus....|(1000,[10,11,12,2..
|message-id: <1qkn...|[message-id:, <1q...|[message-id:, <1q...|(1000,[2,4,17,23,
+--------------------+--------------------+--------------------+----------------
only showing top 20 rows
```

```
from sklearn.metrics import normalized_mutual_info_score as nmi

kmeans_spark_prediction = np.array(predictions.select('prediction').collect()).reshape
kmeans_spark_prediction
```

```
array([0, 1, 1, ..., 1, 1, 0])
```

```
original_labels = np.array(df.select('type_doc').collect())
original_labels = np.where(original_labels == "atheism", 0, 1).reshape(-1)
```

6 analyser les resultats + comparaison avec kmean de sklearn

On remarque que le NMI est assez faible ce qui est facilement explicable car kmeans n'est pas adapté à ce type de données.

```
nmi(original_labels,kmeans_spark_prediction)
```

```
0.005954119237055068
```

transformation des données afin del les adapter à sklearn

```
nomrfeatures_data = np.array(predictions.select('normFeatures').collect()).reshape((19
```

```
nomrfeatures_data.shape
```

```
(1994, 1000)
```

```
features_sklearn = np.array(predictions.select('normFeatures').collect()).shape
```

Lancement de l'algorithme kmean en utilisant le package sklearn

```
%%time
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=2, random_state=0).fit(nomrfeatures_data)
```

```
CPU times: user 1.74 s, sys: 1.04 s, total: 2.79 s
Wall time: 1.61 s
```

```
kmeans.labels_
```

```
array([1, 0, 0, ..., 0, 0, 0], dtype=int32)
```

```
nmi(original_labels,kmeans.labels_)
```

```
0.02242000211585393
```

Remarque

on remarque que dans ce cas ci, sklearn est plus rapide à executer que spark, ceci s'explique car le volume des données n'est pas assez important pour justifier l'utilisation de spark.

Aussi sklearn donne une nmi superieur bien que tout aussi mauvaise pour les méme raisons que celle cité ci-dessus.

7. kmeans unidimensionnel

```
cluster_ids = sc.parallelize([5,4,3,2,2,2,2,1,2,2,2,3,3])
point = sc.parallelize([x for x in range(0,500)])
```

7.a définir la fonction compute centroid

```
from numpy import random

def compute_centroids(points, cluster_ids):

  sum_by_cluster_id = cluster_ids.zip(points).reduceByKey(lambda x,y: x+y)
  count_by_cluster_id = cluster_ids.zip(points).mapValues(lambda x:1).reduceByKey(lamb
  couple = sum_by_cluster_id.join(count_by_cluster_id).mapValues(lambda x: x[0] / x[1]

  return couple
```

7.b éfinir la fonction assign_cluster

7.b.ii dans ce cas ci nous supposons qu'on dispose d'assez de mémoire pour ne pas saturer le driver program.

```
def squared_distances(point_x, vecteur_moy):

 return np.sqrt((np.array(vecteur_moy).reshape(len(vecteur_moy),1) - point_x)**2).resh

def assign_clusters(points, centroids):

  vecteur_moy = centroids.map(lambda x: x[1]).collect() #  dans ce cas ci nous supposo
  return points.map(lambda x: np.argmin(squared_distances(x,vecteur_moy)))
```

7.c implémenter l'etape d'initialisation et itération

```
def kmeans_unidim(points,k=2,max_iter=100):

#  vec =  np.random.rand(k)
  vec = points.takeSample(False, k)
  centroids = sc.parallelize(zip(range(k),vec))

  for i in range(0,max_iter):
    cluster_ids = assign_clusters(points, centroids)
    centroids = compute_centroids(points, cluster_ids)

  cluster_ids = assign_clusters(points, centroids)


  return centroids, cluster_ids
```

```
%%time
centroids, cluster_ids = kmeans_unidim(point,2,10)
```

```
CPU times: user 840 ms, sys: 73.1 ms, total: 913 ms
Wall time: 11.8 s
```

```
print("centroids")
print(centroids.collect())
print("cluster_ids")
print(cluster_ids.collect())
```

```
centroids
[(0, 374.0), (1, 124.0)]
cluster_ids
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
```

8. Kmeans multidimensionnel

modification des fonctions compute_centroid et squared_distances afin de l'adapter à des calcules vectoriel via numpy (np.add, np.divide ...)

```
from numpy import random
def compute_centroids(points, cluster_ids):

  sum_by_cluster_id = cluster_ids.zip(points).reduceByKey(lambda x,y: np.add(x,y).toli
```

```
  count_by_cluster_id = cluster_ids.zip(points).mapValues(lambda x:1).reduceByKey(lamb
  couple = sum_by_cluster_id.join(count_by_cluster_id).mapValues(lambda x: np.divide(x

  return couple


def squared_distances(point_x, vecteur_moy):

 return np.sqrt(np.sum((np.array(vecteur_moy) - point_x)**2,axis=1))

def assign_clusters(points, centroids):

  vecteur_moy = centroids.map(lambda x: x[1]).collect() #supposition qu'on est entrain
  a = points.map(lambda x: np.argmin(squared_distances(x,vecteur_moy)))
  return a



def kmeans_multidim(points,k=2,max_iter=100):

  vec = points.takeSample(False, k)
  centroids = sc.parallelize(zip(range(0,k),vec))

  for i in range(0,max_iter):
    cluster_ids = assign_clusters(points, centroids)
    centroids = compute_centroids(points, cluster_ids)

  cluster_ids = assign_clusters(points, centroids)


  return centroids, cluster_ids
```

```
%%time
points = sc.parallelize([[1,2], [4,5], [7,8], [11,12]])
centroids, cluster_ids = kmeans_multidim(points, 2, 5)
```

```
CPU times: user 456 ms, sys: 44 ms, total: 500 ms
Wall time: 5.85 s
```

```
print("centroids")
print(centroids.collect())
print("cluster_ids")
print(cluster_ids.collect())
```

```
centroids
[(0, [9.0, 10.0]), (1, [2.5, 3.5])]
cluster_ids
[1, 1, 0, 0]
```

9. Spherical Kmean

Modification de la fonction de distance afin d'utiliser la distance du cosine dans assign_cluster, nous utilisons pour cela la fonction cosine_similarity de sklearn. Nous ne voulons donc plus récupérer l'argument minimum dans assign_cluster lors de l'assignation d'un individu à un centre (et donc cluster) mais l'argument maximum (le centre le plus proche et celui disposant de la plus grand cosine similarity)

```
from numpy import random
from sklearn.metrics.pairwise import cosine_similarity

def compute_centroids(points, cluster_ids):

  sum_by_cluster_id = cluster_ids.zip(points).reduceByKey(lambda x,y: np.add(x,y).toli
  count_by_cluster_id = cluster_ids.zip(points).mapValues(lambda x:1).reduceByKey(lamb
  couple = sum_by_cluster_id.join(count_by_cluster_id).mapValues(lambda x: np.divide(x

  return couple

def cosine_distance(point_x, vecteur_moy):

 return cosine_similarity(np.array(vecteur_moy), np.array([point_x])).reshape(-1)

def assign_clusters(points, centroids):

  vecteur_moy = centroids.map(lambda x: x[1]).collect() #supposition qu'on est entrain
  a = points.map(lambda x: np.argmax(cosine_distance(x,vecteur_moy)))
  return a



def skmeans_multidim(points,k=2,max_iter=100):

  vec = points.takeSample(False, k)
  centroids = sc.parallelize(zip(range(0,k),vec))

  for i in range(0,max_iter):
    cluster_ids = assign_clusters(points, centroids)
    centroids = compute_centroids(points, cluster_ids)

  cluster_ids = assign_clusters(points, centroids)


  return centroids, cluster_ids
```

10. analyser les resultat et comparer au package coclust

```
data = nomrfeatures_data[~np.all(nomrfeatures_data == 0, axis=1)]
data= np.delete(data,np.where(~data.any(axis=0))[0], axis=1)
```

```
points = sc.parallelize(data)
```

```
%%time
centroids, cluster_ids = skmeans_multidim(points, 2, 100)
```

```
CPU times: user 10.9 s, sys: 718 ms, total: 11.6 s
Wall time: 5min 16s
```

```
centroids.take(10)
```

```
0.0078074350919276475,
0.0075619370265372675,
0.01960368704515751,
0.009646058788453126,
0.00812276388195918,
0.006386809533106925,
0.008321635695676406,
0.0015281154146850819,
0.0018434773562576007,
0.010566682200711568,
0.004436560611562006,
0.0267209235810126,
0.007126123132208492,
0.010448077671124526,
0.008897967937668347,
0.01006237726414173,
0.004269409107900114,
0.0031560313970156635,
0.008888935776881912,
0.007585367082157545,
0.012504564192860225,
0.007111183474362464,
0.0058205822628306664,
0.0025344102251595385,
0.00864242014639344,
0.005960165932367424,
0.0020046496495780275,
0.005470948407701156,
0.009868934961531448,
0.00785811468007676,
0.012613378864142643,
0.0037138354546692345,
0.019622866069399624,
0.005142883303625457,
0.011895002863529073,
0.006265003956107263,
0.011305979692405892,
0.001287539584601961,
0.005496732165438538,
0.013519098557106257,
0.01300735915577222,
0.0097016879128227
```

```
 0.009701687913283 27,
 0.9365828866946618,
 0.010646709962667915,
 0.007306353736680729,
 0.00996453294468453,
 0.009353075112873713,
 0.01893355485042393,
 0.006941953345337608,
 0.009626747289970284,
 0.014154235229634144,
 0.022259441848937234,
 0.011024289713978772,
 0.019780302837145596,
 0.004436267215577278,
 0.008132451220337119,
 0.016206349641629255,
 0.009214995830377111,
 0.010702502763138487,
```

```
cluster_ids.take(10)
```

```
[0, 1, 1, 1, 1, 1, 1, 1, 1, 1]
```

Skmean avec coclust

```
%%time

from coclust.clustering import SphericalKmeans
import scipy as sp
from scipy import sparse

skmean_model = SphericalKmeans( max_iter=100, n_init=1, random_state=123, weighting=Fa
skmean_model.fit(sparse.csr_matrix(data))
```

```
 == New init ==
iteration: 0
590.3333876247439
iteration: 1
602.385220436727
iteration: 2
608.065900254631
iteration: 3
611.4016323711354
iteration: 4
614.7181148569493
iteration: 5
618.7072412660007
iteration: 6
622.0005947470472
iteration: 7
627.8337578434423
iteration: 8
637.9775150303957
iteration: 9
639.0680004803286
iteration: 10
639.0806199589309
```

```
iteration: 11
CPU times: user 1.61 s, sys: 17.5 ms, total: 1.62 s
Wall time: 1.62 s
```

```
skmean_model.labels_
```

```
[0,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 0,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 0,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 0,
 1,
 1,
 1,
 0,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
```

```
 1,
 1,
 1,
 1,
 1,
```

NMI de notre implémentation

```
nmi(original_labels,cluster_ids.collect())
```

```
0.017111880127824775
```

NMI de l'implémentation coclust

```
nmi(original_labels,skmean_model.labels_)
```

```
0.017111880127824775
```

Remarque :

On peut remarquer qu'en terme de qualité des resutlats sur nos données textuel precedentes en utilisant la NMI, les deux implémentations ont des resultats similaire .

Cependant Coclust est beaucoup plus rapide.

11. MLIB Classification non supervisé

GMM model

```
sample_data = sc.parallelize([[0.0, 0.0, 0.0],
[0.1, 0.1, 0.1],
[0.2, 0.2, 0.2],
[9.0, 9.0, 9.0],
[9.1, 9.1, 9.1],
[9.2, 9.2, 9.2]])
```

```
from numpy import array

from pyspark.mllib.clustering import GaussianMixture, GaussianMixtureModel

# Build the model (cluster the data)
gmm = GaussianMixture.train(sample_data, 2)

# output parameters of model
for i in range(2):
```

```
    print("weight = ", gmm.weights[i], "mu = ", gmm.gaussians[i].mu,
          "sigma = ", gmm.gaussians[i].sigma.toArray())

print("GMM predictions on sample data")
print(gmm.predict(sample_data).collect())
```

```
weight =  0.4988113094096504 mu =  [4.551406083942625,4.551406083942625,4.5514060
 [20.2543053 20.2543053 20.2543053]
 [20.2543053 20.2543053 20.2543053]]
weight =  0.5011886905903497 mu =  [4.648363411531433,4.648363411531433,4.6483634
 [20.25432765 20.25432765 20.25432765]
 [20.25432765 20.25432765 20.25432765]]
predictions on sample data
[0, 0, 0, 1, 1, 1]
```

```
from numpy import array
from math import sqrt

from pyspark.mllib.clustering import KMeans, KMeansModel

# Build the model (cluster the data)
kmean_spark = KMeans.train(sample_data, 2, maxIterations=10, initializationMode="rando

print("GMM predictions on sample data")
print(kmean_spark.predict(sample_data).collect())
```

```
GMM predictions on sample data
[1, 1, 1, 0, 0, 0]
```

12. MLIB classification supervisé

```
from pyspark.mllib.util import MLUtils
```

```
training = spark.read.format("libsvm").load('/content/sample_libsvm_data.txt')
```

Logistic regression

```
from pyspark.ml.classification import LogisticRegression
```

```
lr = LogisticRegression(maxIter=10, regParam=0.3, elasticNetParam=0.8)

# Fit the model
lrModel = lr.fit(training)

# Print the coefficients and intercept for logistic regression
print("Coefficients: " + str(lrModel.coefficients))
```

```
print("Intercept: " + str(lrModel.intercept))

# We can also use the multinomial family for binary classification
mlr = LogisticRegression(maxIter=10, regParam=0.3, elasticNetParam=0.8, family="multin

# Fit the model
mlrModel = mlr.fit(training)

# Print the coefficients and intercepts for logistic regression with multinomial famil
print("Multinomial coefficients: " + str(mlrModel.coefficientMatrix))
print("Multinomial intercepts: " + str(mlrModel.interceptVector))
```

```
Coefficients: (692,[244,263,272,300,301,328,350,351,378,379,405,406,407,428,433,4
Intercept: 0.22456315961250325
Multinomial coefficients: 2 X 692 CSRMatrix
(0,244) 0.0
(0,263) 0.0001
(0,272) 0.0001
(0,300) 0.0001
(0,350) -0.0
(0,351) -0.0
(0,378) -0.0
(0,379) -0.0
(0,405) -0.0
(0,406) -0.0006
(0,407) -0.0001
(0,428) 0.0001
(0,433) -0.0
(0,434) -0.0007
(0,455) 0.0001
(0,456) 0.0001
..
..
Multinomial intercepts: [-0.12065879445860686,0.12065879445860686]
```

Random forest

```
from pyspark.ml import Pipeline
from pyspark.ml.classification import RandomForestClassifier
from pyspark.ml.feature import IndexToString, StringIndexer, VectorIndexer
from pyspark.ml.evaluation import MulticlassClassificationEvaluator

# Index labels, adding metadata to the label column.
data = training
# Fit on whole dataset to include all labels in index.
labelIndexer = StringIndexer(inputCol="label", outputCol="indexedLabel").fit(data)

# Automatically identify categorical features, and index them.
# Set maxCategories so features with > 4 distinct values are treated as continuous.
featureIndexer =\
    VectorIndexer(inputCol="features", outputCol="indexedFeatures", maxCategories=4).f

# Split the data into training and test sets (30% held out for testing)
(trainingData, testData) = data.randomSplit([0.7, 0.3])
```

```
# Train a RandomForest model.
rf = RandomForestClassifier(labelCol="indexedLabel", featuresCol="indexedFeatures", nu

# Convert indexed labels back to original labels.
labelConverter = IndexToString(inputCol="prediction", outputCol="predictedLabel",
                               labels=labelIndexer.labels)

# Chain indexers and forest in a Pipeline
pipeline = Pipeline(stages=[labelIndexer, featureIndexer, rf, labelConverter])

# Train model.  This also runs the indexers.
model = pipeline.fit(trainingData)

# Make predictions.
predictions = model.transform(testData)

# Select example rows to display.
predictions.select("predictedLabel", "label", "features").show(5)

# Select (prediction, true label) and compute test error
evaluator = MulticlassClassificationEvaluator(
    labelCol="indexedLabel", predictionCol="prediction", metricName="accuracy")
accuracy = evaluator.evaluate(predictions)
print("Test Error = %g" % (1.0 - accuracy))

rfModel = model.stages[2]
print(rfModel)  # summary only
```

```
+--------------+-----+--------------------+
|predictedLabel|label|            features|
+--------------+-----+--------------------+
|           0.0|  0.0|(692,[121,122,123...|
|           0.0|  0.0|(692,[123,124,125...|
|           0.0|  0.0|(692,[124,125,126...|
|           0.0|  0.0|(692,[126,127,128...|
|           0.0|  0.0|(692,[126,127,128...|
+--------------+-----+--------------------+
only showing top 5 rows

Test Error = 0
RandomForestClassificationModel: uid=RandomForestClassifier_cc6727bd9514, numTree
```