Department of Computer Science and Engineering

Subject – Big Data
Course Code – UE18CS322
Session – 2020/21

Department of Computer Science and Engineering
PES University Electronic City Campus
Hosur Road, Konappana Agrahara, Electronic City, Bengaluru

Topic : YouTube Data Analysis Using Spark

BY :

Varshit Singh Yadav(PES2201800608)

SparkProject_Youtube_Analysis

This project has been developed on spark framework for analysis of YouTube video data. The analysis of data collected from YouTube provides a brief view of the channels available, the content creators post, and also the likings of the people.

ABSTRACT

There is a tremendous growth and popularity of YouTube. It has the potential to touch billions of lives globally as the no. of YouTube users is growing day by day. YouTube, owned by Google, a video streaming site which has billions of users and 400 hours of videos are being uploaded every minute. Almost billions of videos are watched on YouTube every single day, generating a mammoth amount of data daily. Since YouTube data is generally in unstructured form, there is an increased demand to store, process and analyze such real time Big Data. YouTubers can analyze their own channel performance with YouTube Analytics. But one can not analyze other channels. The proposed system uses spark framework of Hadoop for processing and analyzing real time YouTube datasets. It will help in discovering how competitors are performing on YouTube. One can easily identify what content works best on YouTube. These analytical data can be represented in demographic form which can be used by individuals and organizations for making immediate actionable decisions so as to gain competitive advantages.

CONFIGURATION:

SPARK CONFIGURATION

a. spark.executor.memory = 4g
b. spark.storage.memoryFraction = 1
c. spark.shuffle.memoryFraction = 0.1
d. spark.driver.memory = 4g
e. spark.sql.autoBroadcastJoinThreshold = -1
f. persistence type = MEMORY_AND_DISK
g. version = 3.0.1

SYSTEM CONFIGURATION

1.OS : Ubuntu 20.04(VM)
2.RAM : 2GB
3.Storage : 10GB

OBJECTIVE :

Through **YouTube Analytics**, you can gain insights about your video viewers, find out who they are, what they like, etc. And based on this information, you'll understand what type of content you need to create in order to further engage your audience.

The main aim of this project is to give importance to how data generated from YouTube can be mined and used for making different analysis for companies to focus on targeted, real-time and informative decisions about their products and that can help companies to increase their market values.

Problem Description

In the given Youtube video dataset ,We have to perform following operations ---
* List all the category from text file on which the Youtube video has been Uploaded.
* Find out the number of videos uploaded on each Category.
* Number of uploaded video by each and every user.
* Find Your Top 10 rated Youtube videos.
* Find out the Top 10 videos in each and every Category.
* List of user who have uploaded the video on single and multiple Category.

DATASET PREPROCESSING

Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues. Data preprocessing, if not done, can result in outliers and inconsistencies that can affect the outputs and visualizations.
Some basic preprocessing on the data, we have done are,

1. Removing duplicate rows- The data contained multiple duplicate rows that could cost us the inconsistency in outputs. Since the data is considerably big, they have been removed.
2. Removing missing values (etc. Null/NA values) in the dataset. The dataset consisted a significant 8% of missing, NAN values which have been either removed, or replaced with 0, or with medians.
3. Filtering out all the features from the dataset that are not relevant to our data analysis (etc. X Coordinate, Y Coordinate).
4. Extract date month and year from time-stamp

YOUTUBE DATA DESCRIPTION

YouTube is an American online video-sharing platform headquartered in San Bruno, California YouTube allows users to upload, view, rate, share, add to playlists, report, comment on videos, and subscribe to other users. It offers a wide variety of user-generated and corporate media videos. Available content includes video clips, TV show clips, music videos, short and documentary films, audio recordings, movie trailers, live streams, and other content such as video blogging, short original videos, and educational videos.

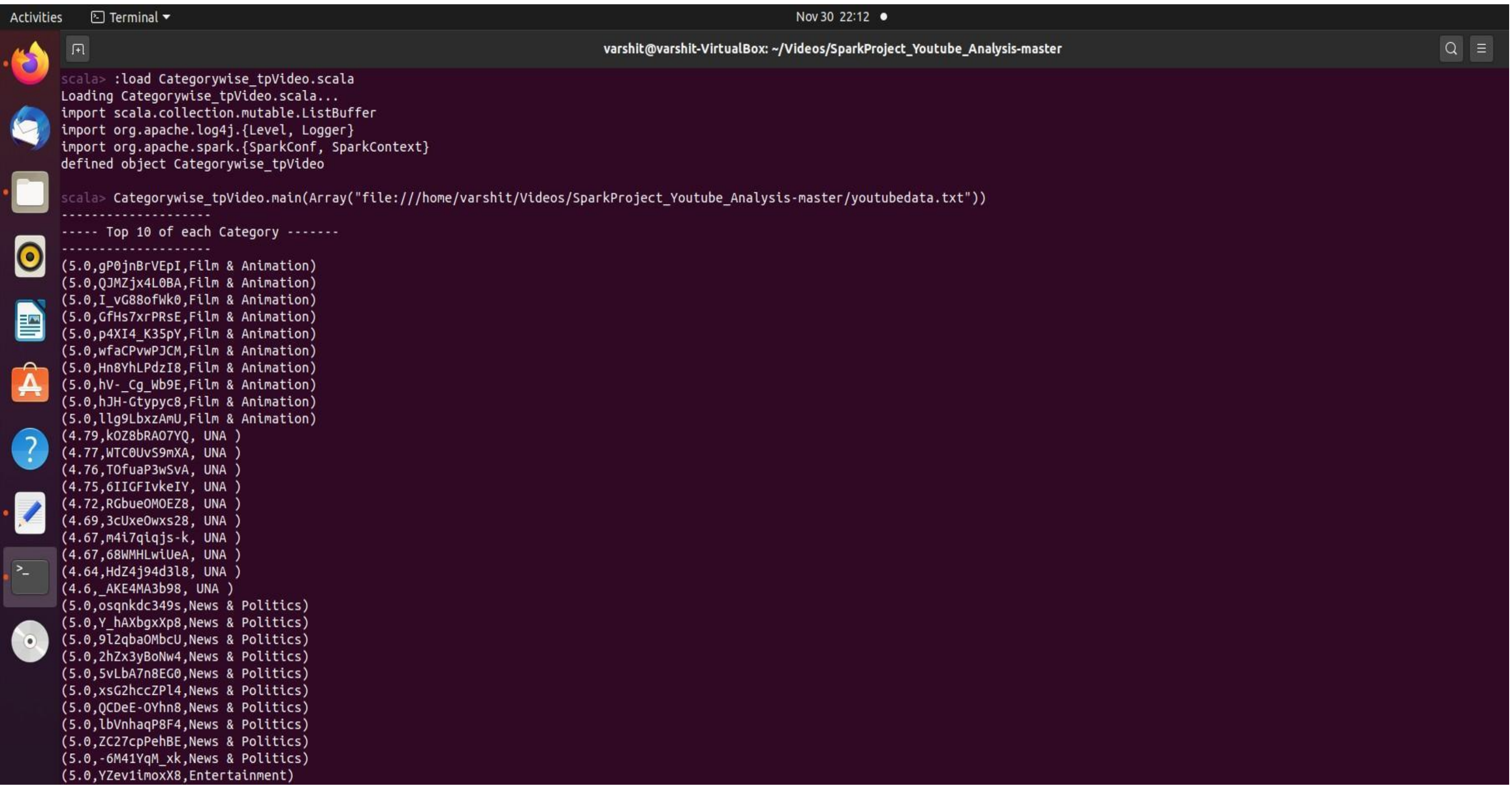
The noticeable columns in which we performed certain operations , in our dataset include video **id of 11 characters** , **uploader of the video** , **time duration of video** , **Category of the video** ,**Length of the video** ,**Number of views for the video** ,**Rating on the video** ,**Number of ratings given for the video** ,**Number of comments done on the videos** ,**Related video ids with the uploaded video** ,**views on it** , **dislikes** etc.

Code and Execution part :

1) **Categorywise_tpVideo (List all the category on which the Youtube video has been Uploaded.)**

```
//package analysis.tube
import
scala.collection.mutable.ListBuffer
import org.apache.log4j.{Level,
Logger}
import org.apache.spark.{SparkConf,
SparkContext} object Categorywise_tpVideo
{
Logger.getLogger("org").setLevel(Level.ER
ROR)

 def main(args: Array[String]) {

 val conf = new
 SparkConf().setMaster("local[3]").setAppName("Count_video_in_Category")
 val sc = new SparkContext(conf)

 val raw_data =
sc.textFile("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-
master/youtubedata.txt")

 val Categories = raw_data.filter(line => line.split("\\t").length >
 4).map { x => val cat = x.split("\\t")(3)
    (cat)
   }
val categories_set = Categories.distinct().collect()
val values = raw_data.filter(line => line.split("\\t").length >
     7).map { line => val lst = line.split("\\t")
     (lst(0), lst(6).toFloat,lst(3))
   }
val
ratingAsKey=values.map{case(x,y,z)=>(y,x,
z)} var top10=ratingAsKey.take(0);
var cat_top10=ratingAsKey.take(0);
categories_set.foreach{ x =>
top10 = ratingAsKey.filter{case(rate,videoid,cats) => cats==x}
                     .sortBy(_._1,ascending = false).take(10)
//top10.foreach(println)
 cat_top10 = cat_top10.union(top10)
}
// Converting tuple map into rdd for storage
purpose val cat_top10RDD =
sc.parallelize(cat_top10,1)
cat_top10RDD.saveAsTextFile("/home/varshit/Videos/SparkProject_Youtu
be_Analysis- master/Category_Top10_videos
println("___________")
println("----- Top 10 of each Category")
println("___________")

cat_top10.foreach(pri
ntln) sc.stop()
}}
```

output:



Activities
Terminal
Nov 30 22:12
varshit@varshit-VirtualBox: ~/Videos/SparkProject_Youtube_Analysis-master

```
scala> :load Categorywise_tpVideo.scala
Loading Categorywise_tpVideo.scala...
import scala.collection.mutable.ListBuffer
import org.apache.log4j.{Level, Logger}
import org.apache.spark.{SparkConf, SparkContext}
defined object Categorywise_tpVideo

scala> Categorywise_tpVideo.main(Array("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-master/youtubedata.txt"))
--------------------
----- Top 10 of each Category -------
--------------------
(5.0,gP0jnBrVEpI,Film & Animation)
(5.0,QJMZjx4L0BA,Film & Animation)
(5.0,I_vG88ofWk0,Film & Animation)
(5.0,GfHs7xrPRsE,Film & Animation)
(5.0,p4XI4_K35pY,Film & Animation)
(5.0,wfaCPvwPJCM,Film & Animation)
(5.0,Hn8YhLPdzI8,Film & Animation)
(5.0,hV-_Cg_Wb9E,Film & Animation)
(5.0,hJH-Gtypyc8,Film & Animation)
(5.0,llg9LbxzAmU,Film & Animation)
(4.79,kOZ8bRAO7YQ, UNA )
(4.77,WTC0UvS9mXA, UNA )
(4.76,TOfuaP3wSvA, UNA )
(4.75,6IIGFIvkeIY, UNA )
(4.72,RGbueOMOEZ8, UNA )
(4.69,3cUxeOwxs28, UNA )
(4.67,m4i7qiqjs-k, UNA )
(4.67,68WMHLwiUeA, UNA )
(4.64,HdZ4j94d3l8, UNA )
(4.6,_AKE4MA3b98, UNA )
(5.0,osqnkdc349s,News & Politics)
(5.0,Y_hAXbgxXp8,News & Politics)
(5.0,9l2qbaOMbcU,News & Politics)
(5.0,2hZx3yBoNw4,News & Politics)
(5.0,5vLbA7n8EG0,News & Politics)
(5.0,xsG2hccZPl4,News & Politics)
(5.0,QCDeE-OYhn8,News & Politics)
(5.0,lbVnhaqP8F4,News & Politics)
(5.0,ZC27cpPehBE,News & Politics)
(5.0,-6M41YqM_xk,News & Politics)
(5.0,YZev1imoxX8,Entertainment)
```

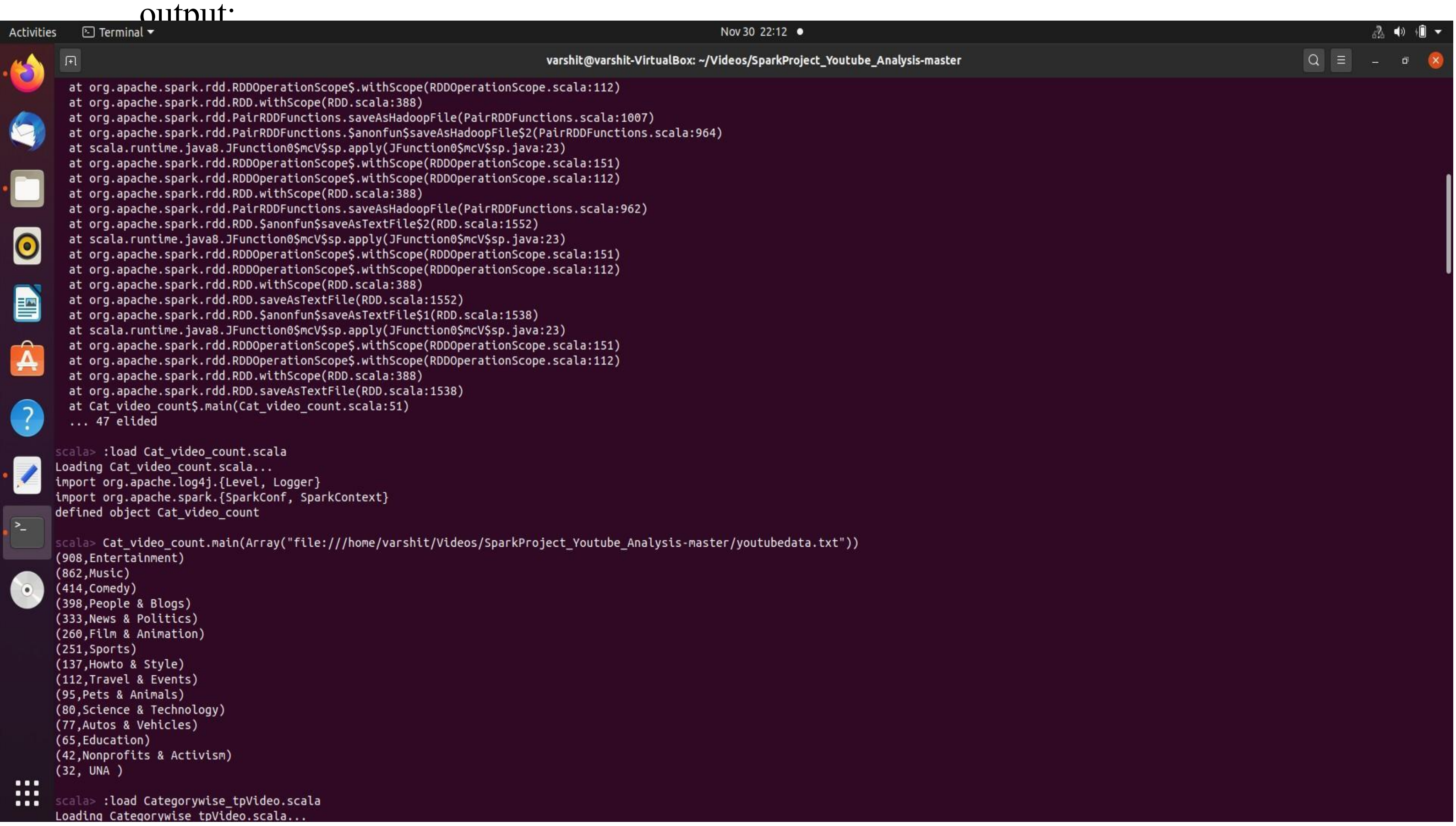
1) Cat_video_count(Find out the number of videos uploaded on each Category.)

```
//package tube.analysis

import org.apache.log4j.{Level, Logger}
import org.apache.spark.{SparkConf,

SparkContext} object Cat_video_count {

Logger.getLogger("org").setLevel(Level.ER
ROR) def main(args:Array[String]){

// val conf = new SparkConf().local("setMaster[3]").setAppName("Count_video_in_Category")
// val sc = new SparkContext.getOrCreate()
val youtube_data = sc.textFile("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-
```

master/youtubedata.txt")

```
 val values =
 youtube_data.filter(line=>line.split("\\t").length>2).map{ line =>
 val lst = line.split("\\t")
  //(video_category
  ,1 ) (lst(3),1)
   }
val                                    reducedData                                    =
values.reduceByKey(_+_).map{case(x,y)=>(y,x)}.sortByKey(false)
reducedData
.saveAsTextFile("/home/varshit/Videos/SparkProject_Youtube_Analysis-
master/Category_video_counts") reducedData .collect().foreach(println)
sc.stop()
}
}
```

output:



Activities
Terminal
Nov 30 22:12
varshit@varshit-VirtualBox: ~/Videos/SparkProject_Youtube_Analysis-master

```
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:112)
  at org.apache.spark.rdd.RDD.withScope(RDD.scala:388)
  at org.apache.spark.rdd.PairRDDFunctions.saveAsHadoopFile(PairRDDFunctions.scala:1007)
  at org.apache.spark.rdd.PairRDDFunctions.$anonfun$saveAsHadoopFile$2(PairRDDFunctions.scala:964)
  at scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:151)
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:112)
  at org.apache.spark.rdd.RDD.withScope(RDD.scala:388)
  at org.apache.spark.rdd.PairRDDFunctions.saveAsHadoopFile(PairRDDFunctions.scala:962)
  at org.apache.spark.rdd.RDD.$anonfun$saveAsTextFile$2(RDD.scala:1552)
  at scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:151)
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:112)
  at org.apache.spark.rdd.RDD.withScope(RDD.scala:388)
  at org.apache.spark.rdd.RDD.saveAsTextFile(RDD.scala:1552)
  at org.apache.spark.rdd.RDD.$anonfun$saveAsTextFile$1(RDD.scala:1538)
  at scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:151)
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:112)
  at org.apache.spark.rdd.RDD.withScope(RDD.scala:388)
  at org.apache.spark.rdd.RDD.saveAsTextFile(RDD.scala:1538)
  at Cat_video_count$.main(Cat_video_count.scala:51)
  ... 47 elided

scala> :load Cat_video_count.scala
Loading Cat_video_count.scala...
import org.apache.log4j.{Level, Logger}
import org.apache.spark.{SparkConf, SparkContext}
defined object Cat_video_count

scala> Cat_video_count.main(Array("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-master/youtubedata.txt"))
(908,Entertainment)
(862,Music)
(414,Comedy)
(398,People & Blogs)
(333,News & Politics)
(260,Film & Animation)
(251,Sports)
(137,Howto & Style)
(112,Travel & Events)
(95,Pets & Animals)
(80,Science & Technology)
(77,Autos & Vehicles)
(65,Education)
(42,Nonprofits & Activism)
(32, UNA )

scala> :load Categorywise_tpVideo.scala
Loading Categorywise_tpVideo.scala...
```

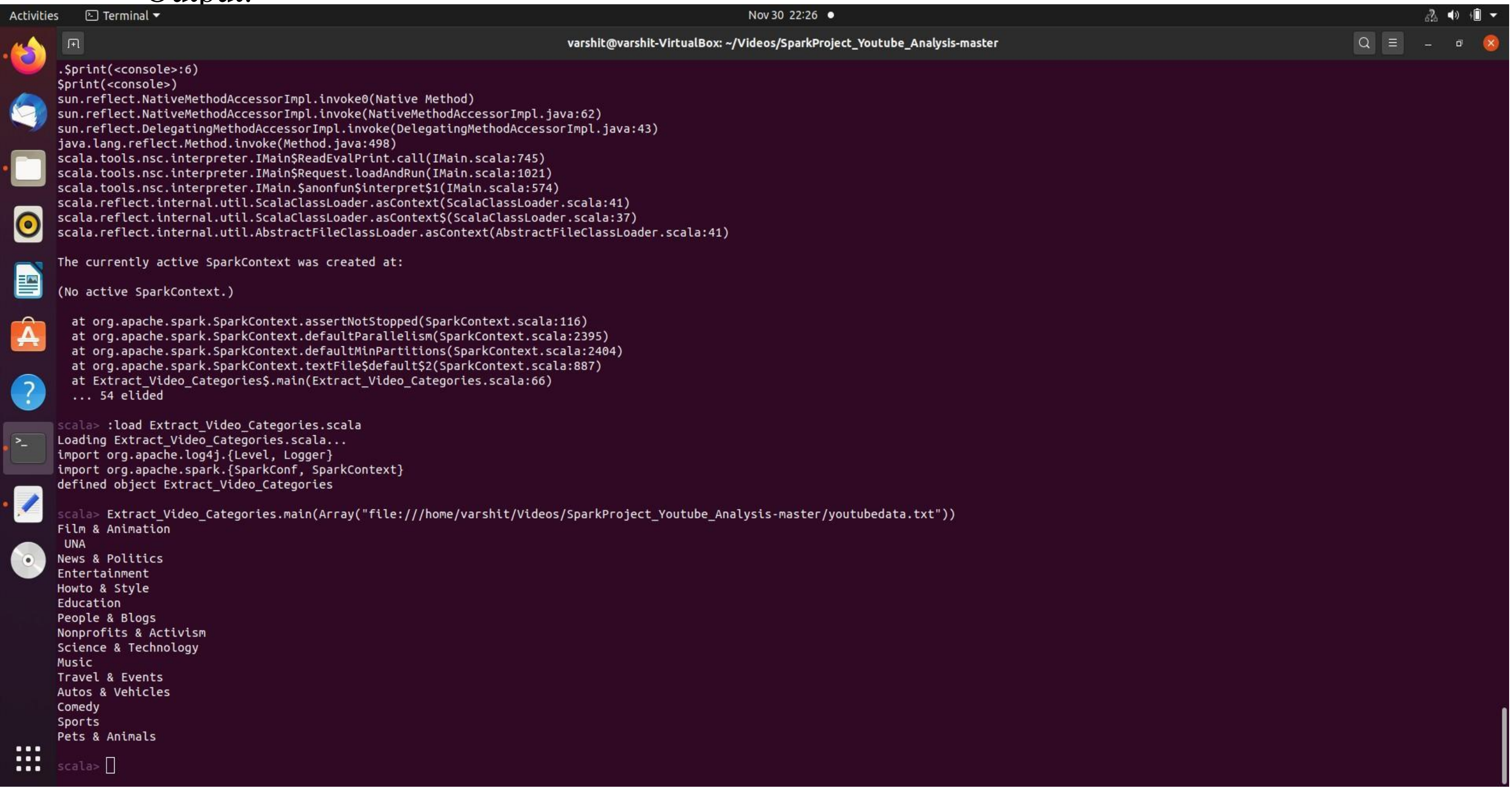
2) Extract Video Categories

```
import org.apache.log4j.{Level, Logger}
import org.apache.spark.{SparkConf, SparkContext}

object Extract_Video_Categories {
Logger.getLogger("org").setLevel(Level.ERROR)

def main(args: Array[String]) {
val conf = new
```

```
SparkConf().setMaster("local[3]").setAppName("Extract_Video_Category") val
sc = new SparkContext(conf)

val raw_data = sc.textFile("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-
master/youtubedata.txt")
```

```
val Categories = raw_data.filter(line => line.split("\\t").length >
4).map { x => val cat = x.split("\\t")(3)
(cat)
   }

Categories.distinct().saveAsTextFile("/home/varshit/Videos/SparkProject_Youtu
be_Analysis- master/Categories")
Categories.distinct().foreach(pr
intln) sc.stop()
  }
  }
```

Output:



Activities
Terminal
Nov 30 22:26
varshit@varshit-VirtualBox: ~/Videos/SparkProject_Youtube_Analysis-master

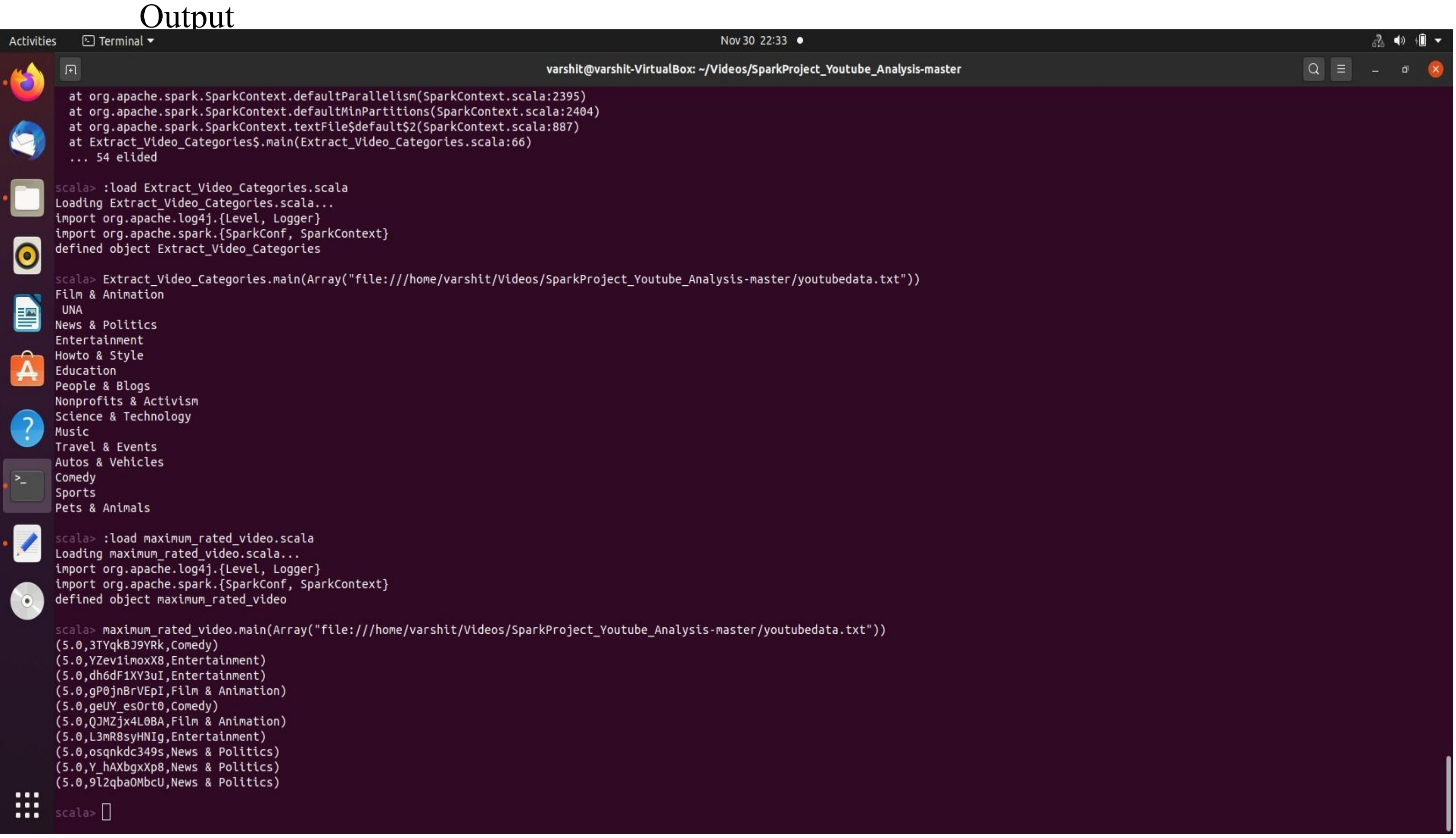
```
.$print(<console>:6)
$print(<console>)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:498)
scala.tools.nsc.interpreter.IMain$ReadEvalPrint.call(IMain.scala:745)
scala.tools.nsc.interpreter.IMain$Request.loadAndRun(IMain.scala:1021)
scala.tools.nsc.interpreter.IMain.$anonfun$interpret$1(IMain.scala:574)
scala.reflect.internal.util.ScalaClassLoader.asContext(ScalaClassLoader.scala:41)
scala.reflect.internal.util.ScalaClassLoader.asContext$(ScalaClassLoader.scala:37)
scala.reflect.internal.util.AbstractFileClassLoader.asContext(AbstractFileClassLoader.scala:41)

The currently active SparkContext was created at:

(No active SparkContext.)

  at org.apache.spark.SparkContext.assertNotStopped(SparkContext.scala:116)
  at org.apache.spark.SparkContext.defaultParallelism(SparkContext.scala:2395)
  at org.apache.spark.SparkContext.defaultMinPartitions(SparkContext.scala:2404)
  at org.apache.spark.SparkContext.textFile$default$2(SparkContext.scala:887)
  at Extract_Video_Categories$.main(Extract_Video_Categories.scala:66)
  ... 54 elided

scala> :load Extract_Video_Categories.scala
Loading Extract_Video_Categories.scala...
import org.apache.log4j.{Level, Logger}
import org.apache.spark.{SparkConf, SparkContext}
defined object Extract_Video_Categories

scala> Extract_Video_Categories.main(Array("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-master/youtubedata.txt"))
Film & Animation
 UNA
News & Politics
Entertainment
Howto & Style
Education
People & Blogs
Nonprofits & Activism
Science & Technology
Music
Travel & Events
Autos & Vehicles
Comedy
Sports
Pets & Animals

scala> 
```

3) Maximum rated video

```
import org.apache.log4j.{Level, Logger}
import org.apache.spark.{SparkConf,

SparkContext} object maximum_rated_video

{

Logger.getLogger("org").setLevel(Level.ER
ROR) def main(args: Array[String]) {

val conf = new
SparkConf().setMaster("local[3]").setAppName("Top_Rated_Video") val
sc = new SparkContext(conf)
```

```
val youtube_raw_data =
sc.textFile("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-master/youtubedata.txt")
val values = youtube_raw_data.filter(line => line.split("\\t").length > 7).map { line =>
val lst = line.split("\\t")
 (lst(0),
 lst(6).toFloat,lst(3))
    }

val ratingAsKey=values.map{case(x,y,z)=>(y,x,z)}
val top10 = ratingAsKey.sortBy(_._1,ascending =
false).take(10) val top10Rdd = sc.parallelize(top10,1)
   top10Rdd.foreach(println)
top10Rdd.saveAsTextFile("/home/varshit/Videos/SparkProject_Youtub
e_Analysis- master/topRatedVideos")
sc.stop()
  }
}
```

Output


Activities
Terminal
Nov 30 22:33
varshit@varshit-VirtualBox: ~/Videos/SparkProject_Youtube_Analysis-master

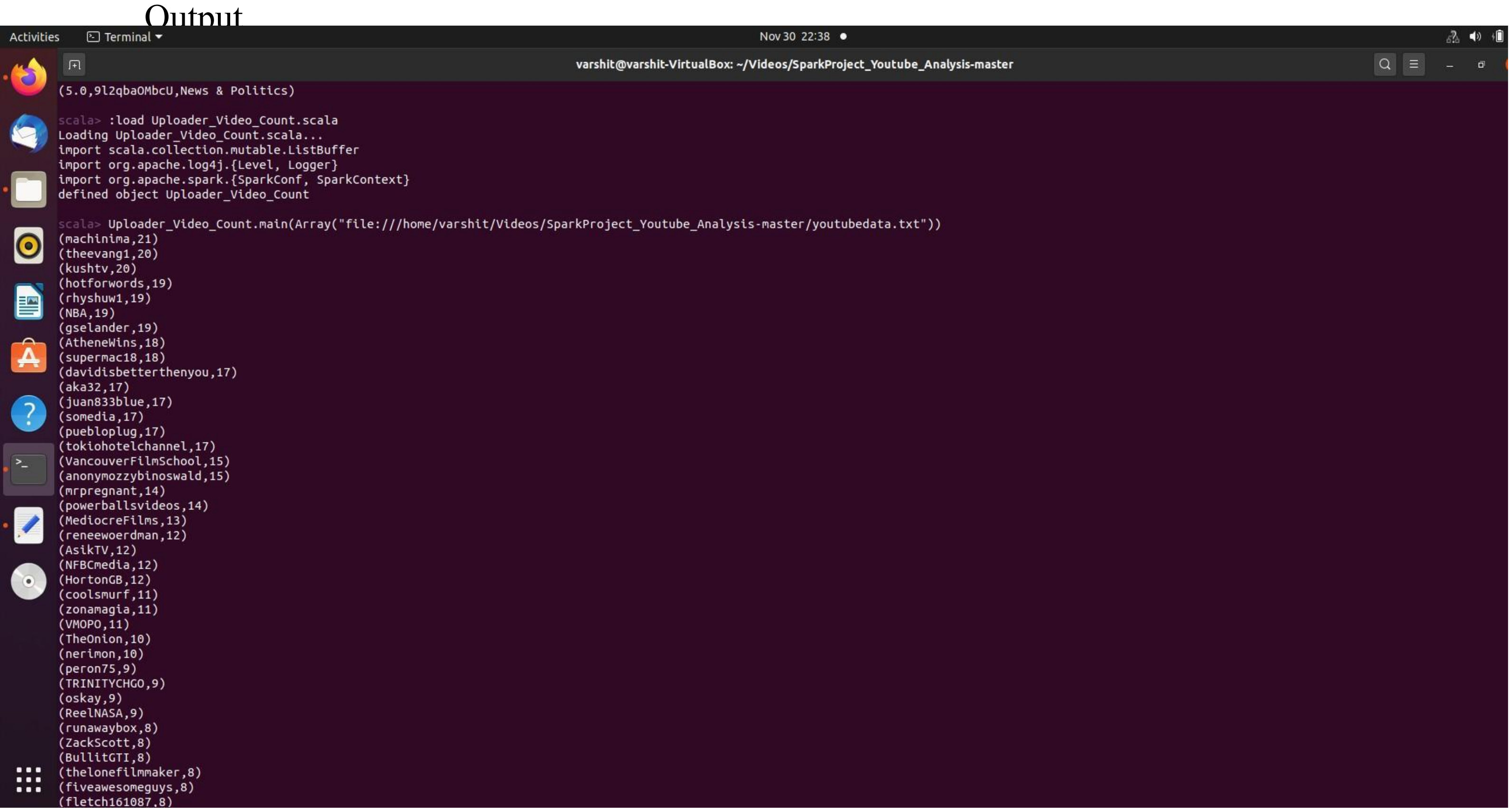
```
  at org.apache.spark.SparkContext.defaultParallelism(SparkContext.scala:2395)
  at org.apache.spark.SparkContext.defaultMinPartitions(SparkContext.scala:2404)
  at org.apache.spark.SparkContext.textFile$default$2(SparkContext.scala:887)
  at Extract_Video_Categories$.main(Extract_Video_Categories.scala:66)
  ... 54 elided

scala> :load Extract_Video_Categories.scala
Loading Extract_Video_Categories.scala...
import org.apache.log4j.{Level, Logger}
import org.apache.spark.{SparkConf, SparkContext}
defined object Extract_Video_Categories

scala> Extract_Video_Categories.main(Array("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-master/youtubedata.txt"))
Film & Animation
 UNA
News & Politics
Entertainment
Howto & Style
Education
People & Blogs
Nonprofits & Activism
Science & Technology
Music
Travel & Events
Autos & Vehicles
Comedy
Sports
Pets & Animals

scala> :load maximum_rated_video.scala
Loading maximum_rated_video.scala...
import org.apache.log4j.{Level, Logger}
import org.apache.spark.{SparkConf, SparkContext}
defined object maximum_rated_video

scala> maximum_rated_video.main(Array("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-master/youtubedata.txt"))
(5.0,3TYqkBJ9YRk,Comedy)
(5.0,YZev1imoxX8,Entertainment)
(5.0,dh6dF1XY3uI,Entertainment)
(5.0,gP0jnBrVEpI,Film & Animation)
(5.0,geUY_esOrt0,Comedy)
(5.0,QJMZjx4L0BA,Film & Animation)
(5.0,L3mR8syHNIg,Entertainment)
(5.0,osqnkdc349s,News & Politics)
(5.0,Y_hAXbgxXp8,News & Politics)
(5.0,9l2qbaOMbcU,News & Politics)

scala> 
```

4) **Uploader Video Count**

```
import
scala.collection.mutable.ListB
uffer import
org.apache.log4j.{Level,
Logger}
import org.apache.spark.{SparkConf, SparkContext}
```

```
object Uploader_Video_Count {
Logger.getLogger("org").setLevel(Level.ERROR)

 def main(args: Array[String]) {
 val conf = new SparkConf().setMaster("local[2]").setAppName("Number
 of Uploaded Video by User") val sc = new SparkContext(conf)
 val raw_data =
 sc.textFile("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-master/
 youtubedata.txt") val user_list = raw_data.filter(line => line.split("\\t").length
 >1).map(line => (line.split("\\t")(1),1) )
 val video_count_ofuser =
user_list.reduceByKey(_+_).sortBy(_._2,ascending = false)
video_count_ofuser.saveAsTextFile("/home/varshit/Videos/Spark
Project_Youtube_Analysis- master/User_Video_Count")
 video_count_ofuser.foreach(pr
  intln) sc.stop()
}
}
```

Output



Activities
Terminal
Nov 30 22:38
varshit@varshit-VirtualBox: ~/Videos/SparkProject_Youtube_Analysis-master

```
(5.0,9l2qbaOMbcU,News & Politics)

scala> :load Uploader_Video_Count.scala
Loading Uploader_Video_Count.scala...
import scala.collection.mutable.ListBuffer
import org.apache.log4j.{Level, Logger}
import org.apache.spark.{SparkConf, SparkContext}
defined object Uploader_Video_Count

scala> Uploader_Video_Count.main(Array("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-master/youtubedata.txt"))
(machinima,21)
(theevang1,20)
(kushtv,20)
(hotforwords,19)
(rhyshuw1,19)
(NBA,19)
(gselander,19)
(AtheneWins,18)
(supermac18,18)
(davidisbetterthenyou,17)
(aka32,17)
(juan833blue,17)
(somedia,17)
(puebloplug,17)
(tokiohotelchannel,17)
(VancouverFilmSchool,15)
(anonymozzybinoswald,15)
(mrpregnant,14)
(powerballsvideos,14)
(MediocreFilms,13)
(reneewoerdman,12)
(AsikTV,12)
(NFBCmedia,12)
(HortonGB,12)
(coolsmurf,11)
(zonamagia,11)
(VMOPO,11)
(TheOnion,10)
(nerimon,10)
(peron75,9)
(TRINITYCHGO,9)
(oskay,9)
(ReelNASA,9)
(runawaybox,8)
(ZackScott,8)
(BullitGTI,8)
(thelonefilmmaker,8)
(fiveawesomeguys,8)
(fletch161087,8)
```

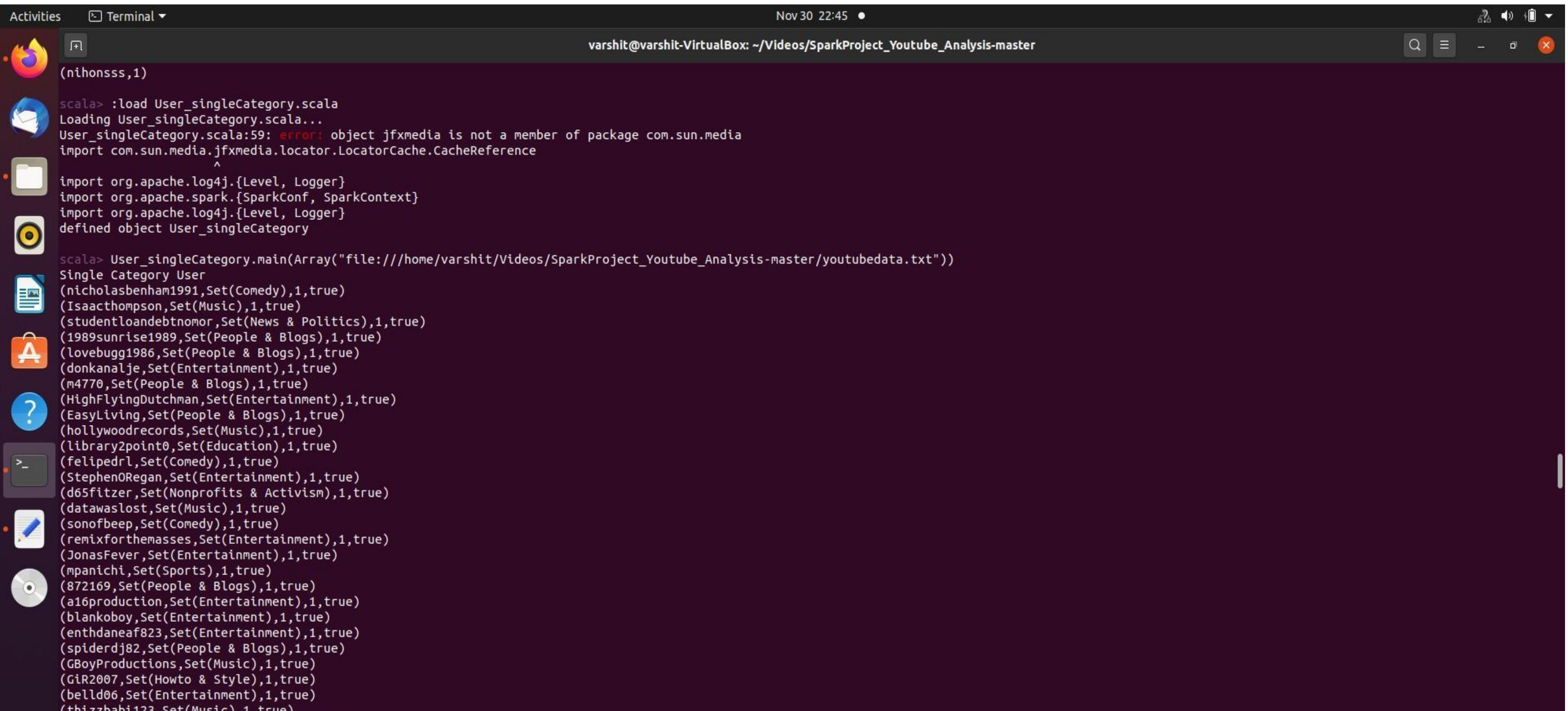
5)User single Category

```
import
com.sun.media.jfxmedia.locator.LocatorCache.CacheRefere
nce import org.apache.log4j.{Level, Logger}
import org.apache.spark.{SparkConf,
SparkContext} import
org.apache.log4j.{Level, Logger}
object User_singleCategory {
Logger.getLogger("org").setLevel(Level.ERROR)

def main(args: Array[String]): Unit = {
```

```
val conf = new SparkConf().setMaster("local[3]").setAppName("user video in
single category") val sc = new SparkContext(conf)
val raw_data = sc.textFile("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-
master/youtubedata.txt")

val userand_Cateory = raw_data.filter(line => line.split("\\t").length >
3).map { x => val video_info = x.split("\\t")
 (video_info(1),video_info(3))
   }
```

```
var category_lst =
List[Any]() var
category_set =
Set[Any]()
val user_category_grp = userand_Cateory.groupByKey()

val user_category_detail = user_category_grp.map{ case(userId,category_Buffer) =>

  category_lst = Nil
  category_Buffer.foreach{ category
  => category_lst = category ::
  category_lst
  }
  category_set = category_lst.toSet
  (userId,category_set,category_set.size,if(category_set.size==1)
  true else false)
   }
   val user_upload_onecategory = user_category_detail.filter( record =>
   record._4==true) val user_upload_mulcategory =
   user_category_detail.filter(record => record._4==false) println("Single
   Category User")
   user_upload_onecategory.foreach(pr
   intln) println("Multiple Category
   User")
   user_upload_mulcategory.foreach(p
   rintln) sc.stop()
}
}
```

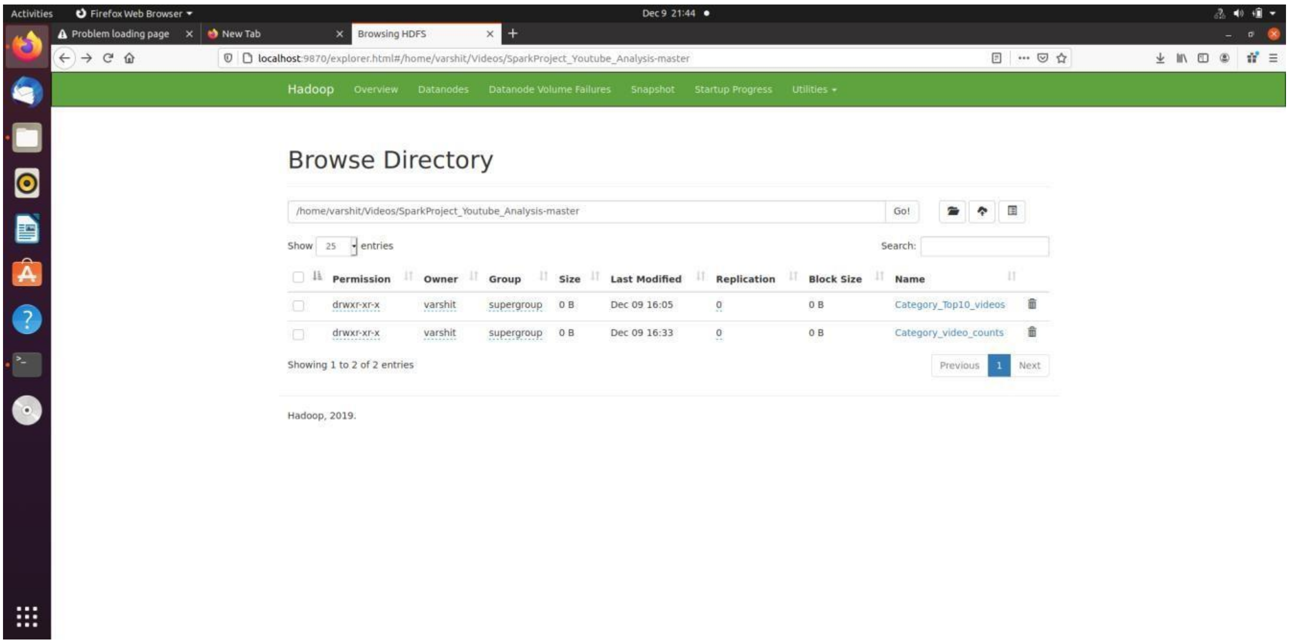
Output



Activities
Terminal
Nov 30 22:45
varshit@varshit-VirtualBox: ~/Videos/SparkProject_Youtube_Analysis-master

```
(nihonsss,1)

scala> :load User_singleCategory.scala
Loading User_singleCategory.scala...
User_singleCategory.scala:59: error: object jfxmedia is not a member of package com.sun.media
import com.sun.media.jfxmedia.locator.LocatorCache.CacheReference
                     ^
import org.apache.log4j.{Level, Logger}
import org.apache.spark.{SparkConf, SparkContext}
import org.apache.log4j.{Level, Logger}
defined object User_singleCategory

scala> User_singleCategory.main(Array("file:///home/varshit/Videos/SparkProject_Youtube_Analysis-master/youtubedata.txt"))
Single Category User
(nicholasbenham1991,Set(Comedy),1,true)
(Isaacthompson,Set(Music),1,true)
(studentloandebtnomor,Set(News & Politics),1,true)
(1989sunrise1989,Set(People & Blogs),1,true)
(lovebugg1986,Set(People & Blogs),1,true)
(donkanalje,Set(Entertainment),1,true)
(m4770,Set(People & Blogs),1,true)
(HighFlyingDutchman,Set(Entertainment),1,true)
(EasyLiving,Set(People & Blogs),1,true)
(hollywoodrecords,Set(Music),1,true)
(library2point0,Set(Education),1,true)
(felipedrl,Set(Comedy),1,true)
(StephenORegan,Set(Entertainment),1,true)
(d65fitzer,Set(Nonprofits & Activism),1,true)
(datawaslost,Set(Music),1,true)
(sonofbeep,Set(Comedy),1,true)
(remixforthemasses,Set(Entertainment),1,true)
(JonasFever,Set(Entertainment),1,true)
(mpanichi,Set(Sports),1,true)
(872169,Set(People & Blogs),1,true)
(a16production,Set(Entertainment),1,true)
(blankoboy,Set(Entertainment),1,true)
(enthdaneaf823,Set(Entertainment),1,true)
(spiderdj82,Set(People & Blogs),1,true)
(GBoyProductions,Set(Music),1,true)
(GiR2007,Set(Howto & Style),1,true)
(belld06,Set(Entertainment),1,true)
(thizzbabi123,Set(Music),1,true)
```

HDFS Input And Output:



localhost:9870/explorer.html#/home/varshit/Videos/SparkProject_Youtube_Analysis-master

Browse Directory

/home/varshit/Videos/SparkProject_Youtube_Analysis-master Go!

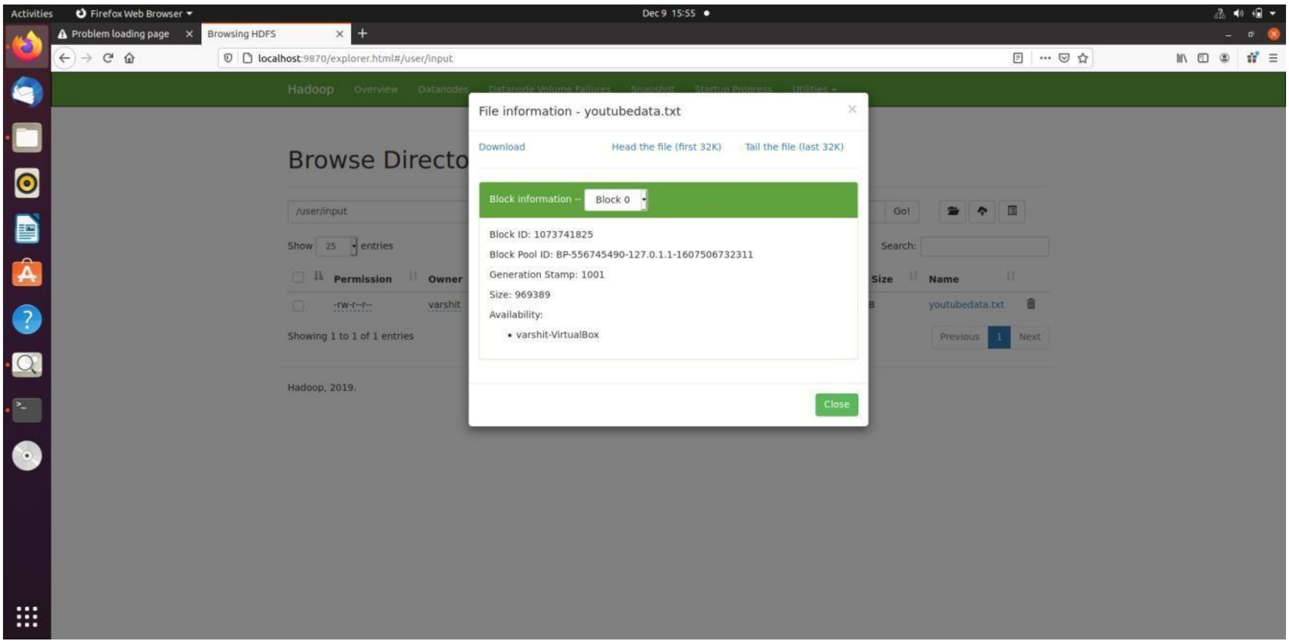
Show 25 entries Search:

□	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
□	drwxr-xr-x	varshit	supergroup	0 B	Dec 09 16:05	0	0 B	Category_Top10_videos
□	drwxr-xr-x	varshit	supergroup	0 B	Dec 09 16:33	0	0 B	Category_video_counts

Showing 1 to 2 of 2 entries

Previous 1 Next

Hadoop, 2019.



localhost:9870/explorer.html#/user/input

Browse Directo

/user/input Go!

Show 25 entries Search:

□	Permission	Owner	Size	Name
□	-rw-r--r--	varshit	B	youtubedata.txt

Showing 1 to 1 of 1 entries

Previous 1 Next

Hadoop, 2019.

File information - youtubedata.txt

Download Head the file (first 32K) Tail the file (last 32K)

Block information -- Block 0

Block ID: 1073741825

Block Pool ID: BP-556745490-127.0.1.1-1607506732311

Generation Stamp: 1001

Size: 969389

Availability:

- varshit-VirtualBox

Close

-

CONCLUSION:

In this paper, we presented our findings for measuring, analysing, and comparing key aspects of YouTube trending videos. Our study has been based on monitoring the viewership and related statistics. Since trending videos are declared as such just several hours after they are uploaded, we are able to analyse trending videos' time-series across critical and sufficiently-long durations of their lifecycle. We presented an extensive data-driven analysis on the lifecycle of trending videos to the best of our knowledge. We also presented the basic characteristics of trending videos popularity over their lifetime. In addition, we analysed the profile of users who upload trending videos. Furthermore, we conducted a significance testing to conduct this analysis.

Our directional-relationship analysis provided a deeper insight onto the viewership pattern of different categories of trending videos. Key findings of our study include the following. Trending videos and their channels have clear distinct statistical attributes when compared to other YouTube content that has not been labelled as trending. The viewership of nearly all trending videos has some values in our dataset.

Our results also reveal a highly asymmetric directional-relationship among different categories of trending videos. Our directionality analysis also shows a clear pattern of viewership toward poplar categories, whereas some categories tend to be isolated with little evidence of transitions among them

REFERENCES

1. Webster, John. "MapReduce: Simplified Data Processing on Large Clusters", "Search Storage", 2004. Retrieved on 25 March 2013. https://static.googleusercontent.com/media/research.g oogle.com/en//archive/mapreduce-osdi04.pdf

2. Bibliography: Big Data Analytics: Methods and Applications by Saumyadipta Pyne, B.L.S. Prakasa Rao, S.B. Rao

3. YOUTUBE COMPANY STATISTICS.https://www.statisticbrain.com/youtube-statistics/

4. Youtube.com @2017. YouTube for media. https://www.youtube.com/yt/about/press/

5. Big data;Wikipedia https://en.wikipedia.org/wiki/Big_data

6. Kallerhoff,Phillip. ―Big Data and Credit Unions: Machine Learning in Member Transactions https://filene.org/assets/pdfreports/301_Kallerhoff_M achine_Learning.pdf

7. Marr,Barnard.―Why only one of the 5 Vs of big data really matters http://www.ibmbigdatahub.com/blog/why-only-one-5- vs- big-data-really-matters