# Big Data Hadoop Training

Project: Twitter Sentiment Analysis Solution

**You need to stream data from twitter and you need to perform sentiment analysis on the data.**

**Follow the below blog and perform sentiment analysis using Mapreduce/pig**

**https://acadgild.com/blog/sentiment-analysis-on-tweets-using-afinn-dictionary/**

**Submit the screen shots of the final results with the Source code.**

**Step-by-step procedure followed in performing twitter sentiment analysis:**

* We will be collecting real-time tweets from Twitter using Flume.

**Streaming Twitter Data Using Flume**

* Create the flume configuration code as shown below.
* Change the twitter api keys with the keys of our Twitter App
* Now, we have to decide which keywords tweet data to be collected from the twitter application, then change the keywords in the TwitterAgent.sources.Twitter.keywords command.

In our example, we are fetching tweet data related to Hadoop, election, sports, cricket, Big data and Bahubali.

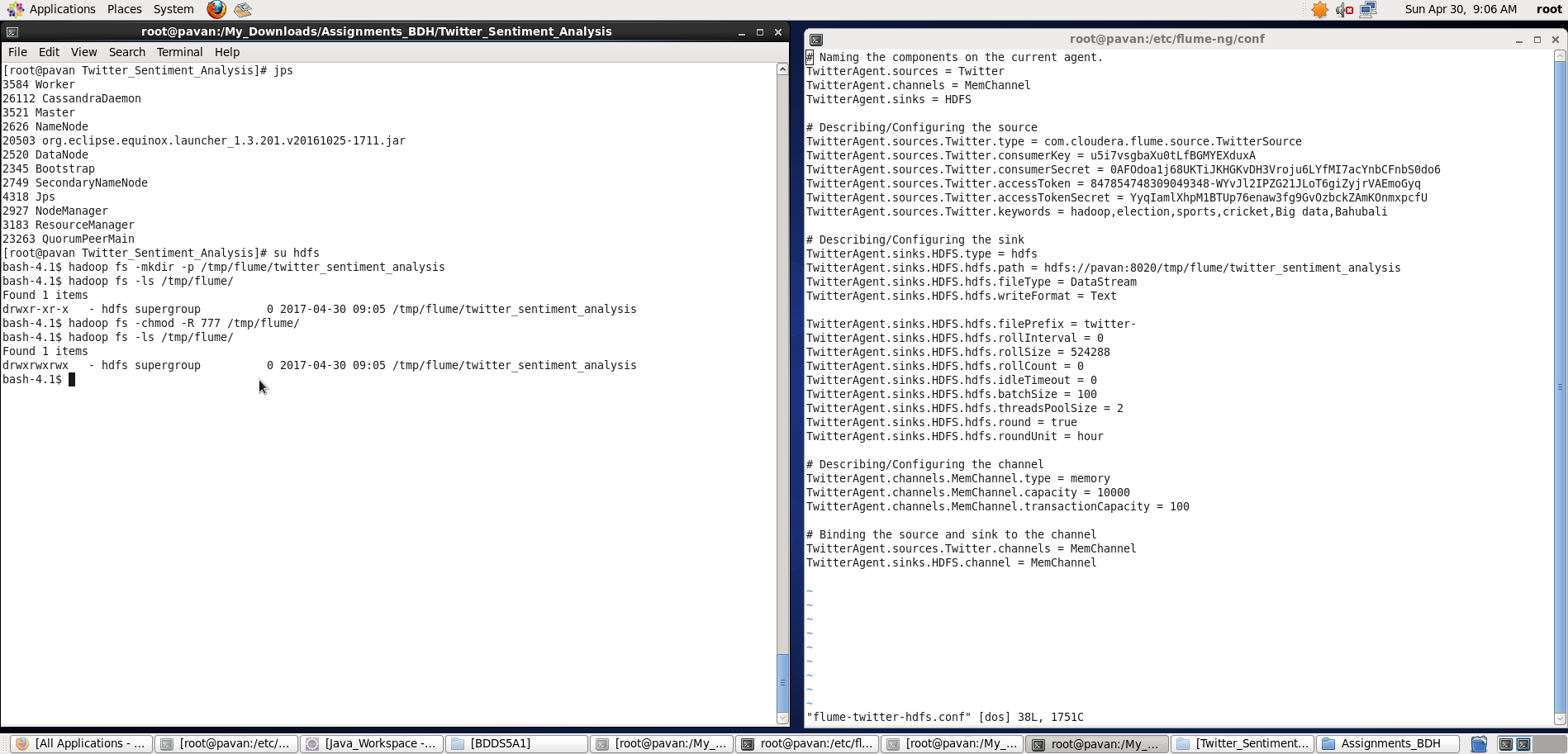
* Open a new terminal and start all the Hadoop daemons, before running the flume command to fetch the twitter data.

Use the ‘jps’ command to see the running Hadoop daemons.

* Create a new directory inside HDFS path, where the Twitter tweet data should be stored.

**hadoop fs –mkdir –p /tmp/flume/twitter\_sentiment\_analysis**

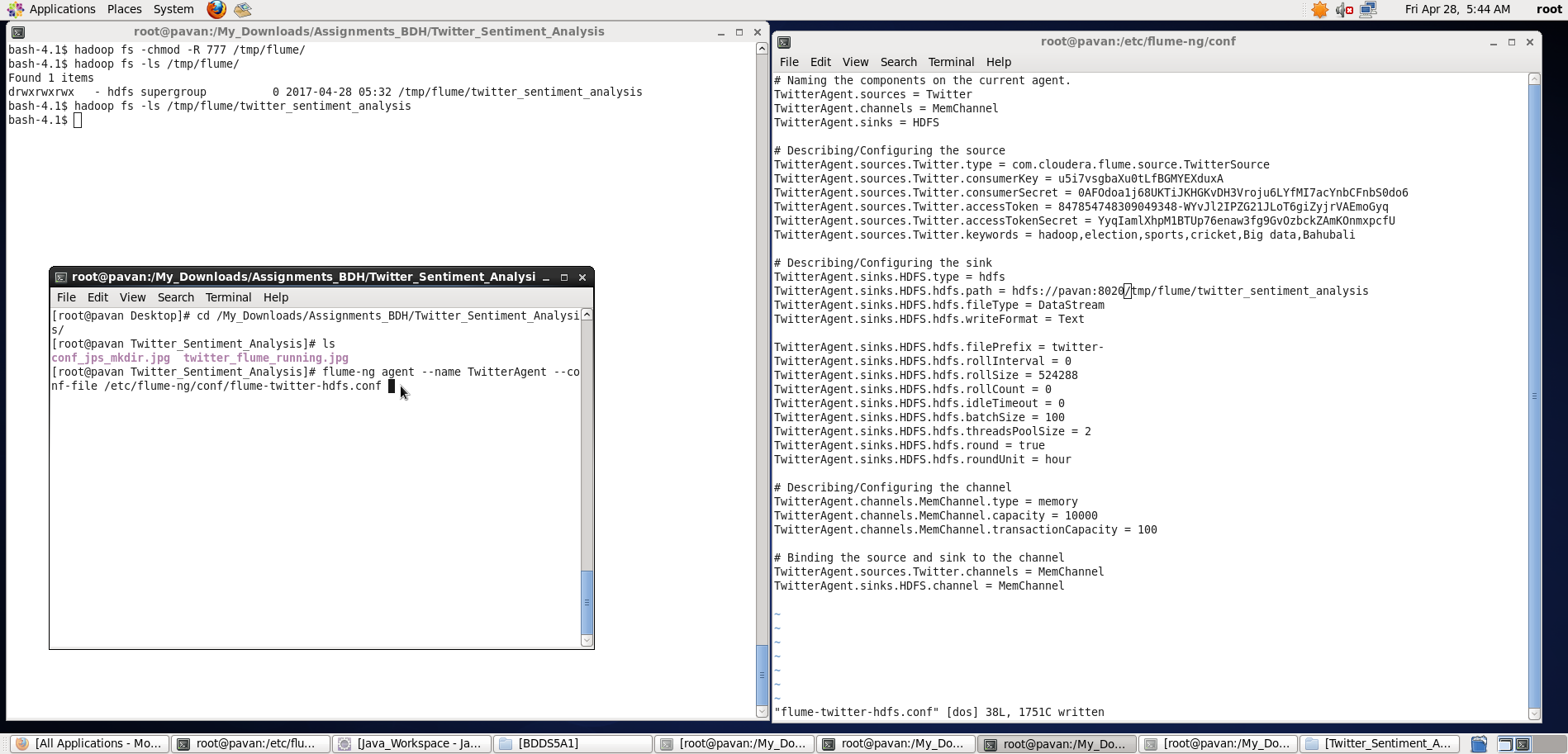
* **Note: Give appropriate read/write permissions for FLUME to write into the HDFS.**



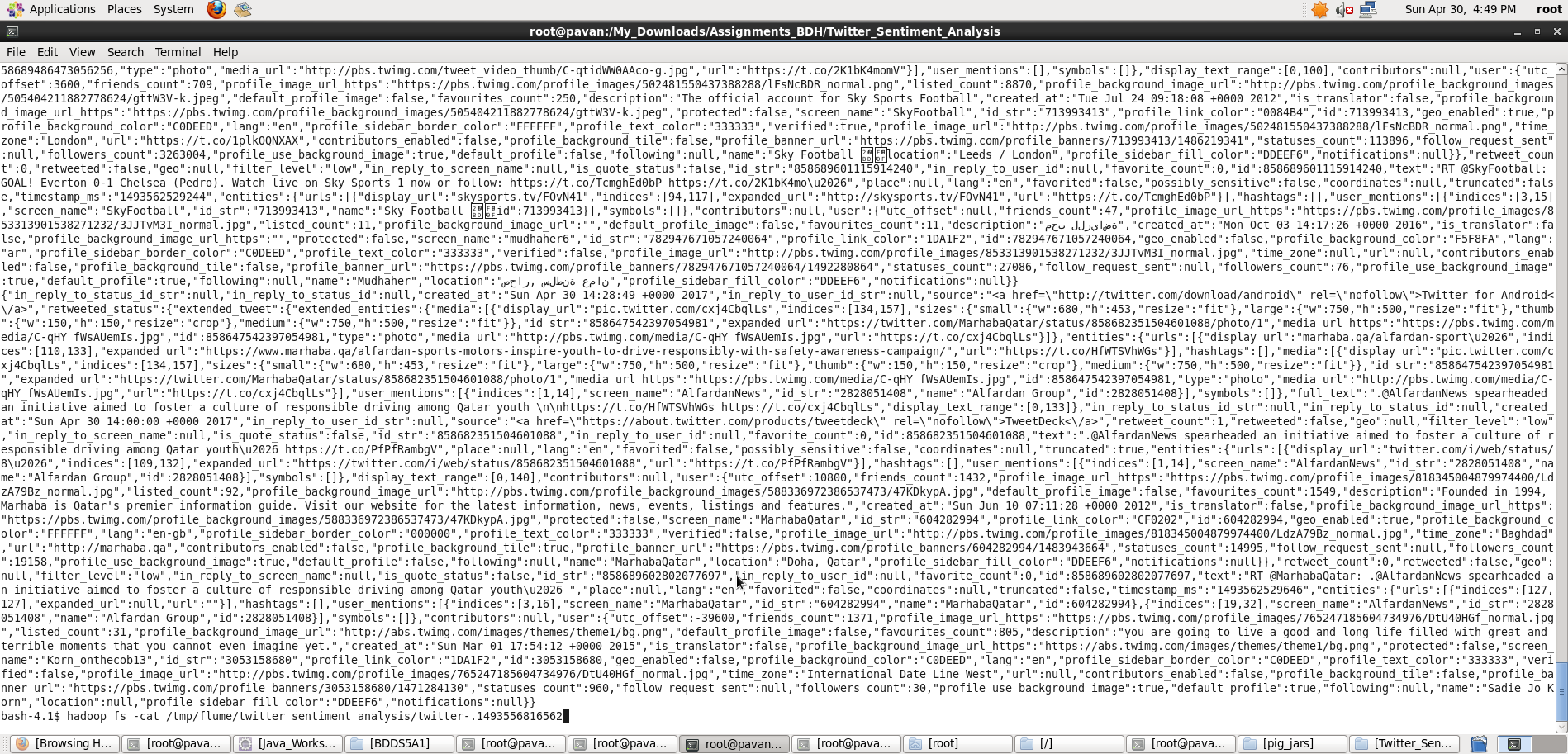
* For fetching data from Twitter, Use the below command to fetch the twitter tweet data into the HDFS cluster path.

**flume-ng agent --name TwitterAgent –conf-file /etc/flume-ng/conf/flume-twitter-hdfs.conf**

The above command will start fetching data from Twitter and steams it into the HDFS given path



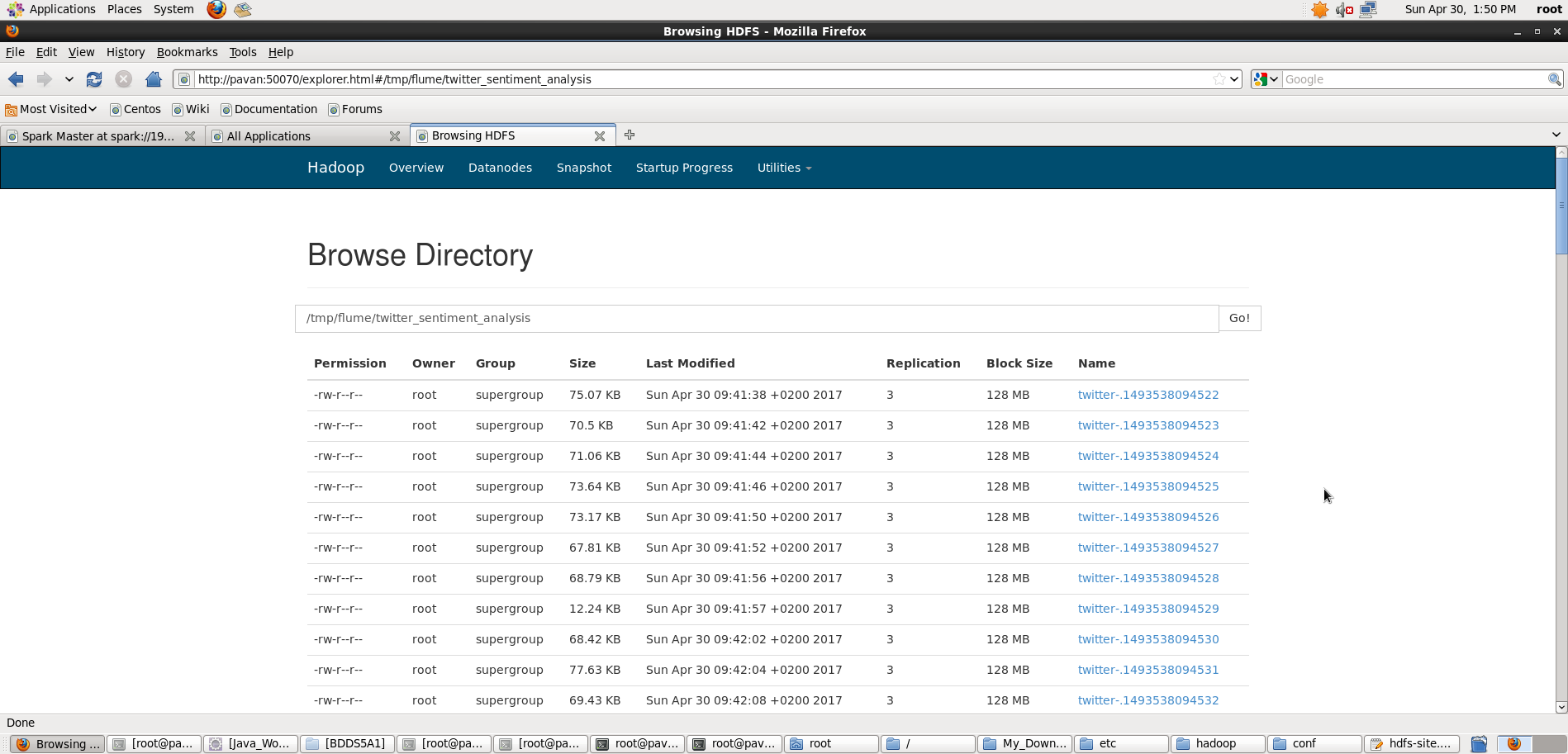
* Once, the tweet data started streaming it into the given HDFS path we can use ‘Ctrl+c’ command to stop the streaming process.
* To check the contents of the tweet data we can use the following command:
* **hadoop fs –ls /tmp/flume/twitter\_sentiment\_analysis**
* We can use the *‘cat’* command to display the tweet  data inside the /tmp/flume/twitter\_sentiment\_analysis/twitter-149\* path.
* **hadoop fs –cat /tmp/flume/twitter\_sentiment\_analysis/<flumeData file name>**



* We can observe from the above image that we have successfully fetched twitter data into our HDFS cluster directory.  Once the tweets have been successfully stored in your database, we can manipulate the tweet data to fit the needs of our sentimental analysis.

**Sentimental Analysis on Twitter Data stored in HDFS**

* All the real-time tweets is kept it the location ‘**/tmp/flume/twitter\_sentiment\_analysis**‘ directory in HDFS. You can refer to the below screen shot for the same.



* The data from Twitter is in ‘Json’ format, so a Pig JsonLoader is required to load the data into Pig.

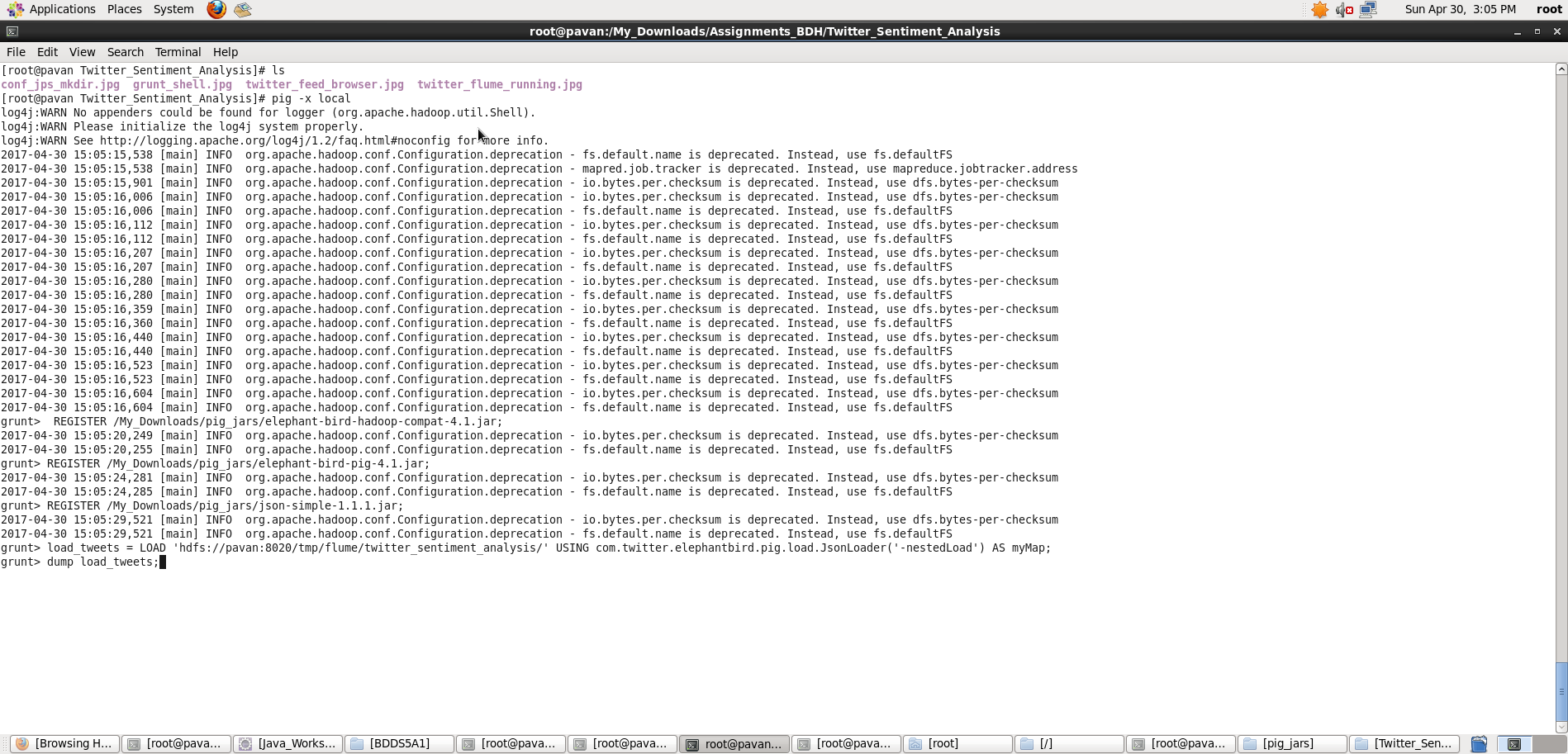
Register the downloaded jars in pig by using the below commands

REGISTER '/My\_Downloads/pig\_jars/elephant-bird-hadoop-compat-4.1.jar';

REGISTER '/My\_Downloads/pig\_jars/elephant-bird-pig-4.1.jar';

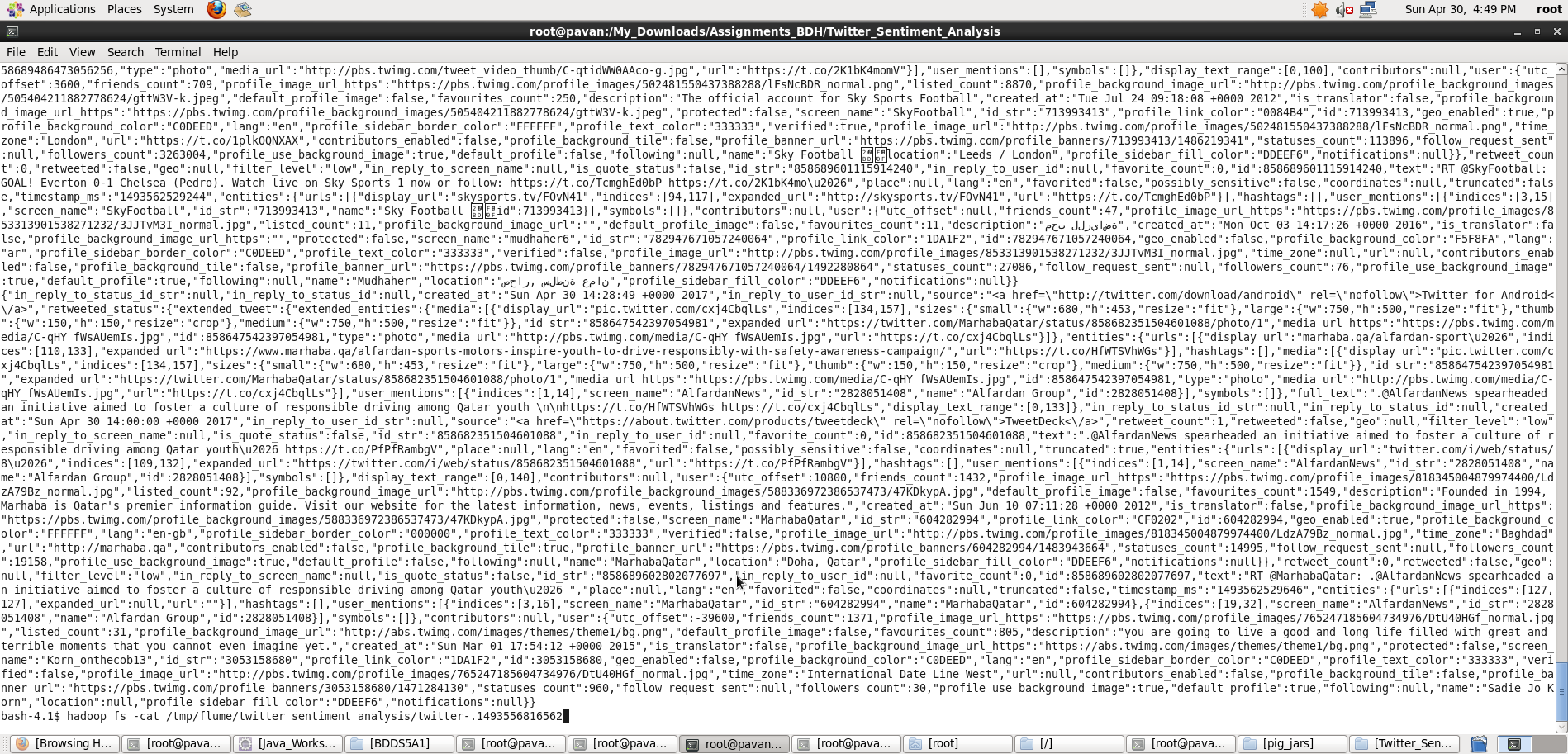
REGISTER '/My\_Downloads/pig\_jars/json-simple-1.1.1.jar';

**Note**: I am running Pig in Local Mode



* After registering the required jars, we can now write a Pig script to perform Sentiment Analysis

Below is a sample tweets collected for this purpose



* The tweets are in nested Json format and consists of map data types. We need to load the tweets using JsonLoader which supports maps, so we are using **elephant bird JsonLoader**to load the tweets.
* Below is the first Pig statement required to load the tweets into Pig:

load\_tweets = LOAD 'hdfs://pavan:8020/tmp/flume/twitter\_sentiment\_analysis/' USING com.twitter.elephantbird.pig.load.JsonLoader('-nestedLoad') AS myMap;

When we dump the above relation, we can see that all the tweets got loaded successfully.



* Now, we shall extract the **id**and the **tweet text** from the above tweets. The Pig statement necessary to perform this is:

extract\_details = FOREACH load\_tweets GENERATE myMap#'id' as id,myMap#'text' as text;



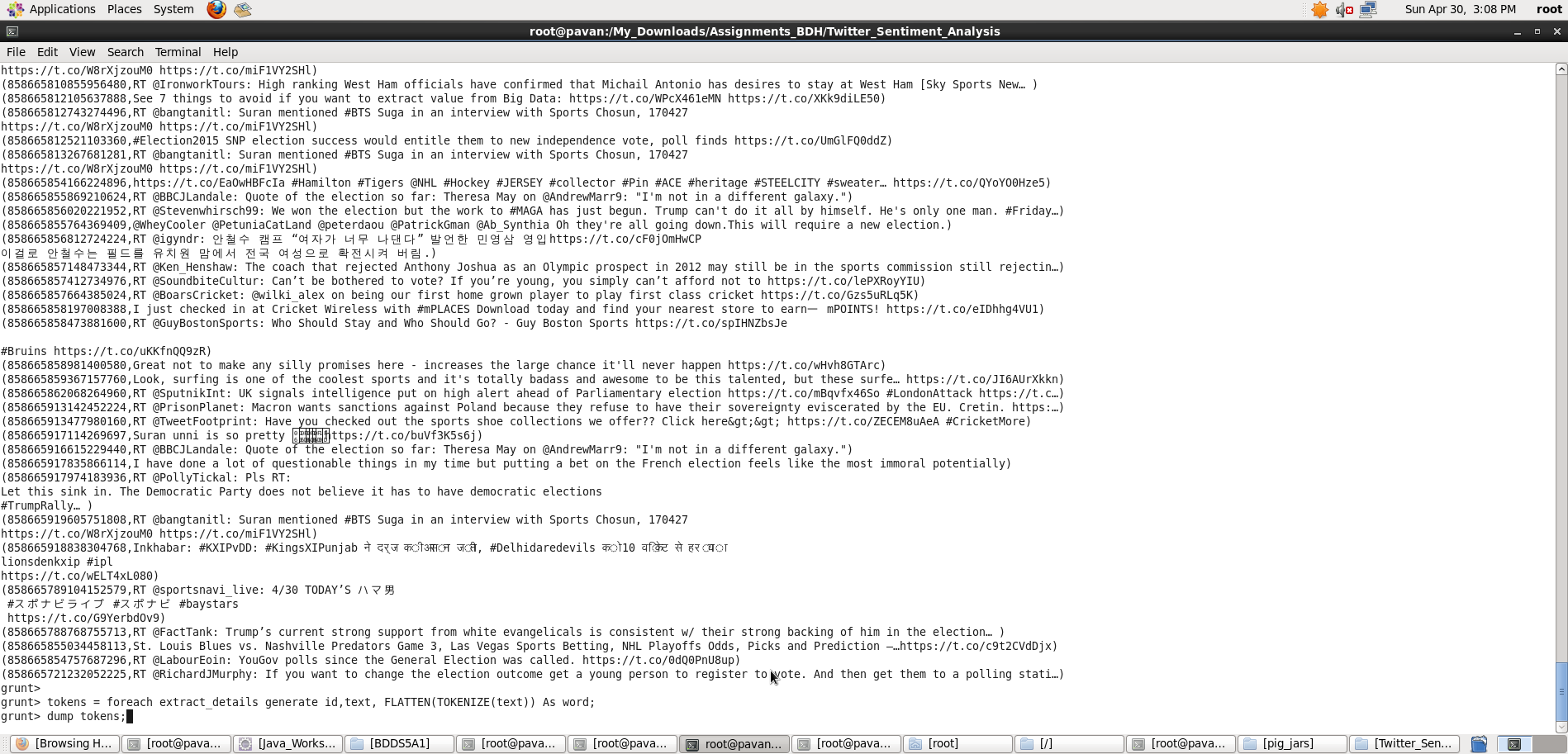
We can see the extracted **id** and **tweet text**from the tweets in the below screen shot.



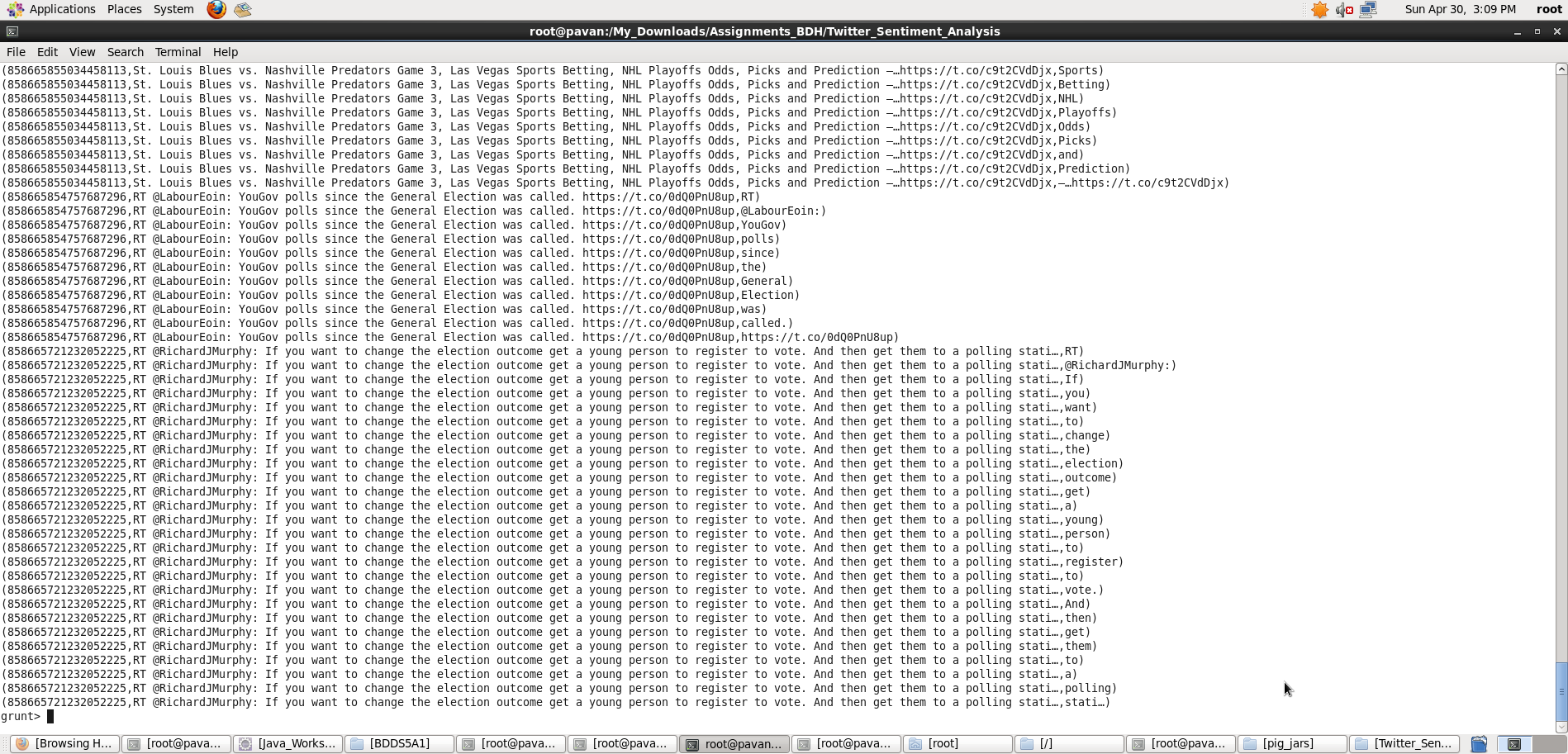
We have the tweet id and the tweet text in the relation named as **extract\_details**.

* Now, we shall extract the words from the text using the TOKENIZE key word in Pig.

tokens = foreach extract\_details generate id,text, FLATTEN(TOKENIZE(text)) As word;

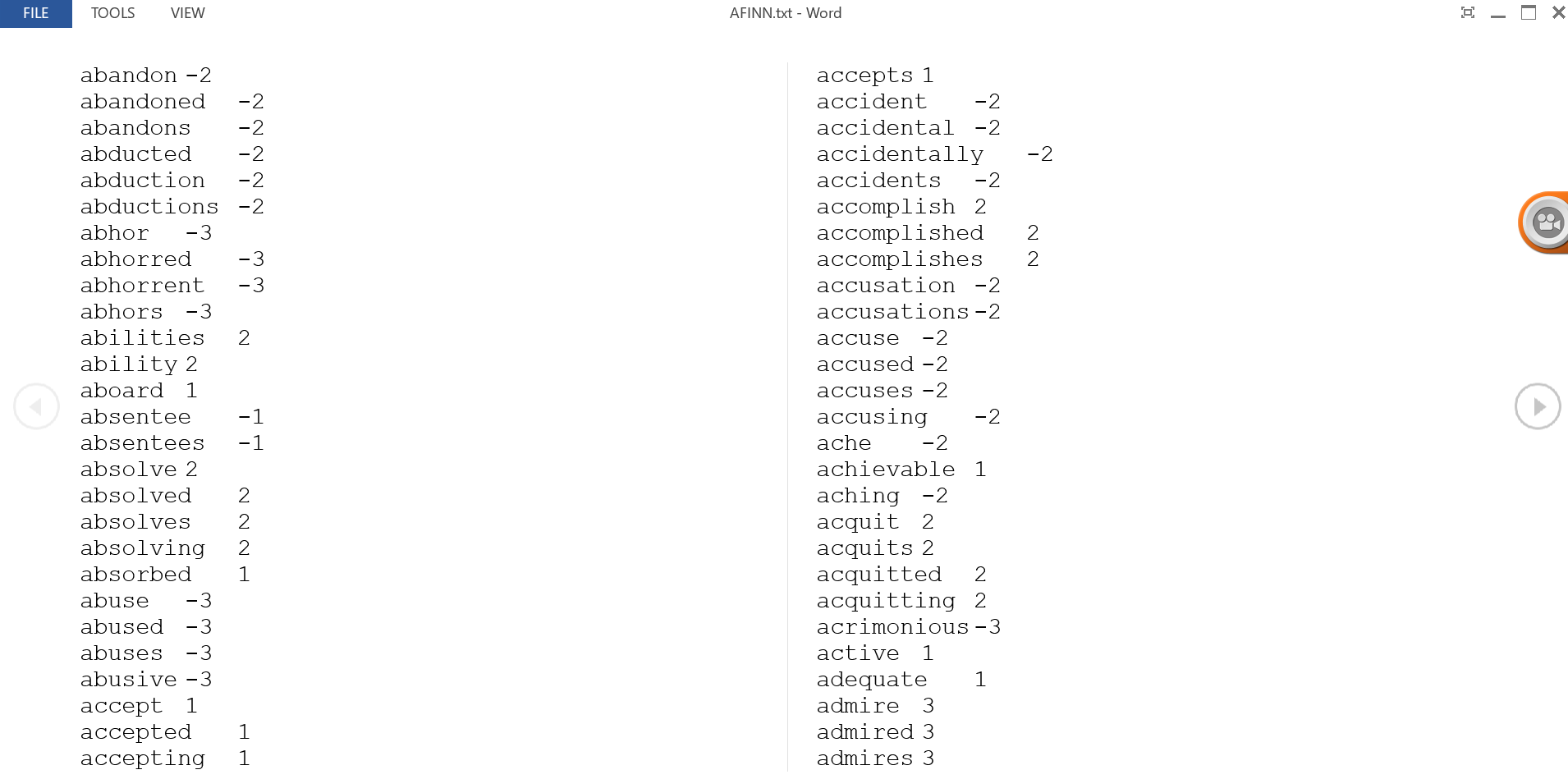


From the below screen shot, we can see that the text got divided into words.



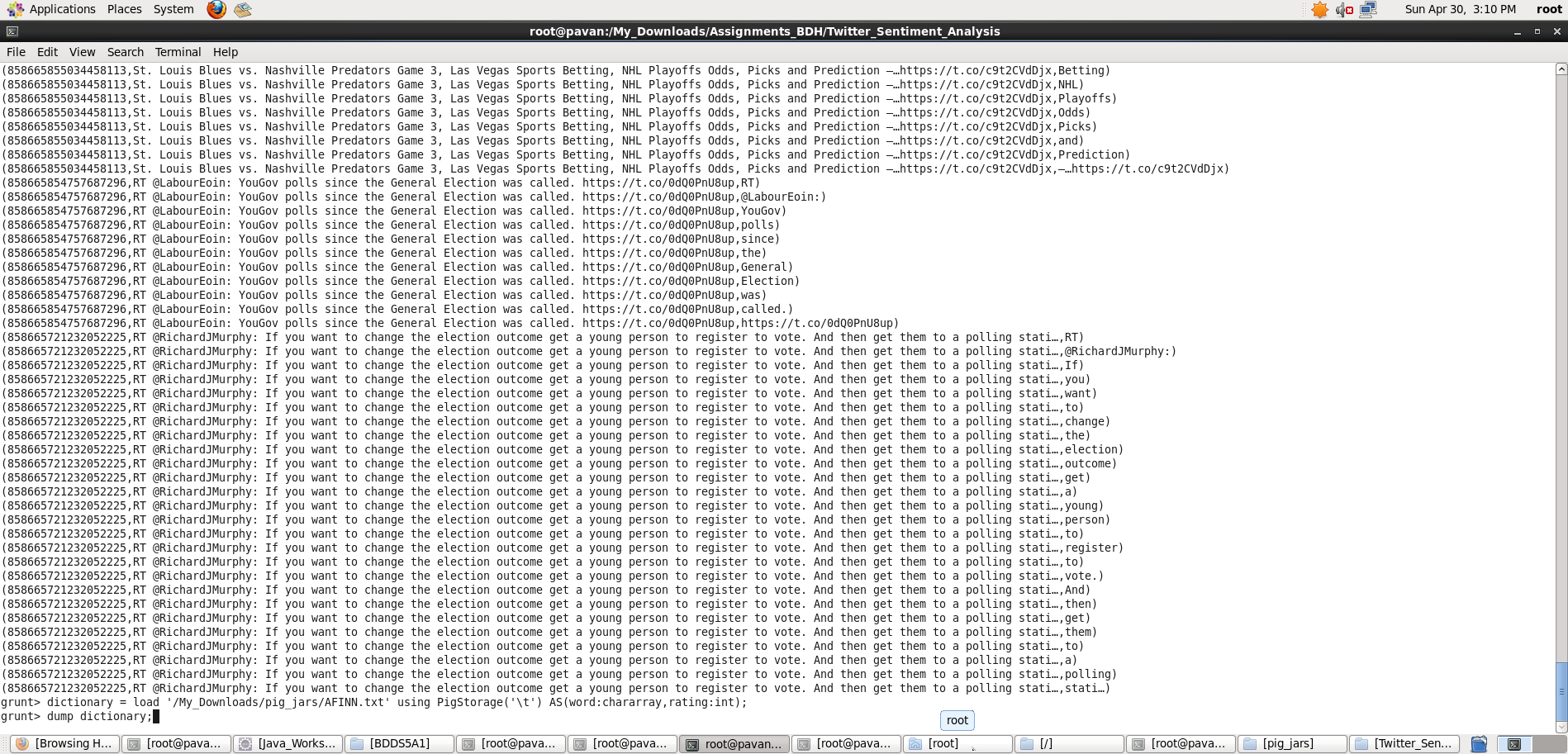
* Now, we have to analyze the Sentiment for the tweet by using the words in the text.
* We will rate the word as per its meaning from +5 to -5 using the dictionary AFINN.

The AFINN is a dictionary which consists of 2500 words which are rated from +5 to -5 depending on their meaning.

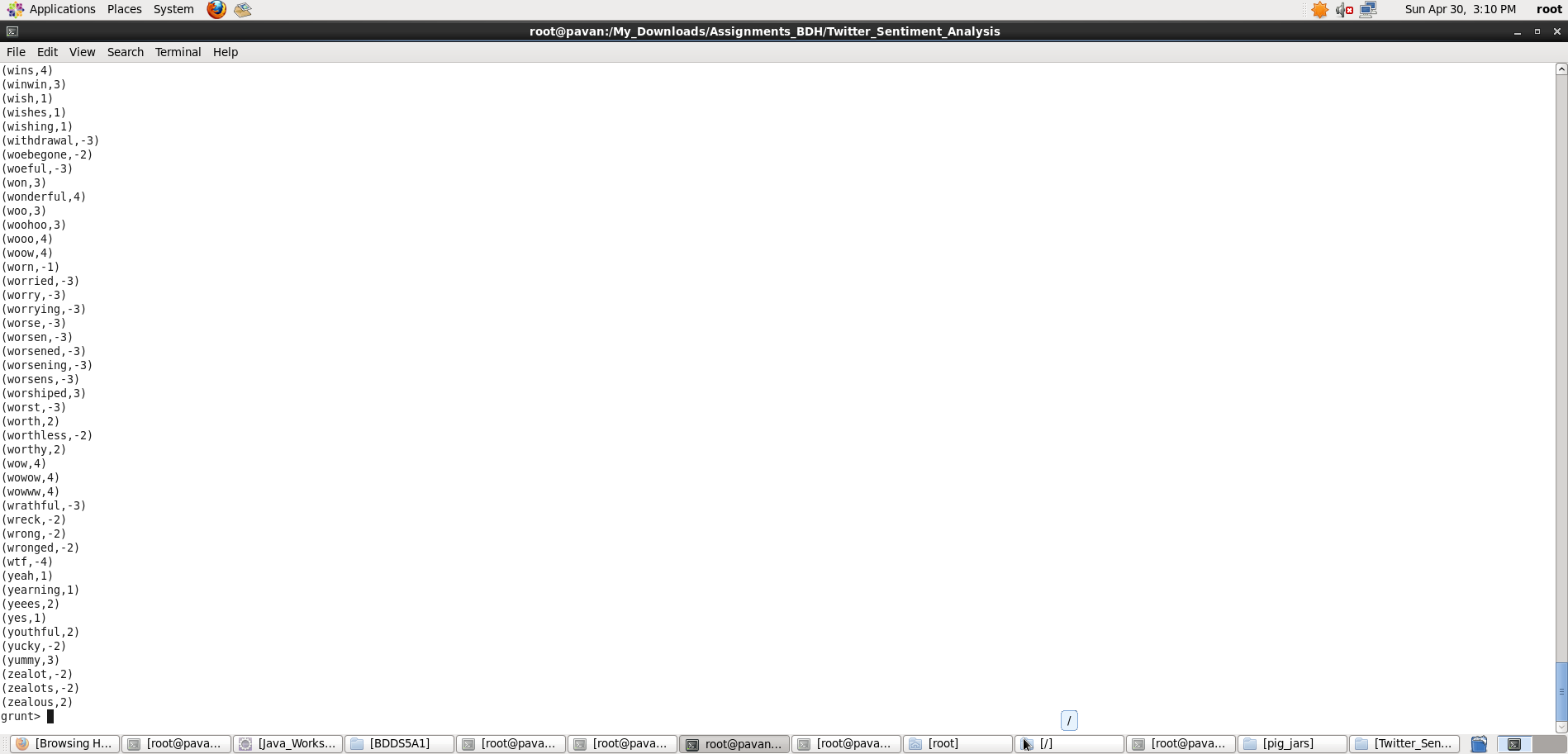


* We will load the dictionary into pig by using the below statement:

dictionary = load '/AFINN.txt' using PigStorage('\t') AS(word:chararray,rating:int);



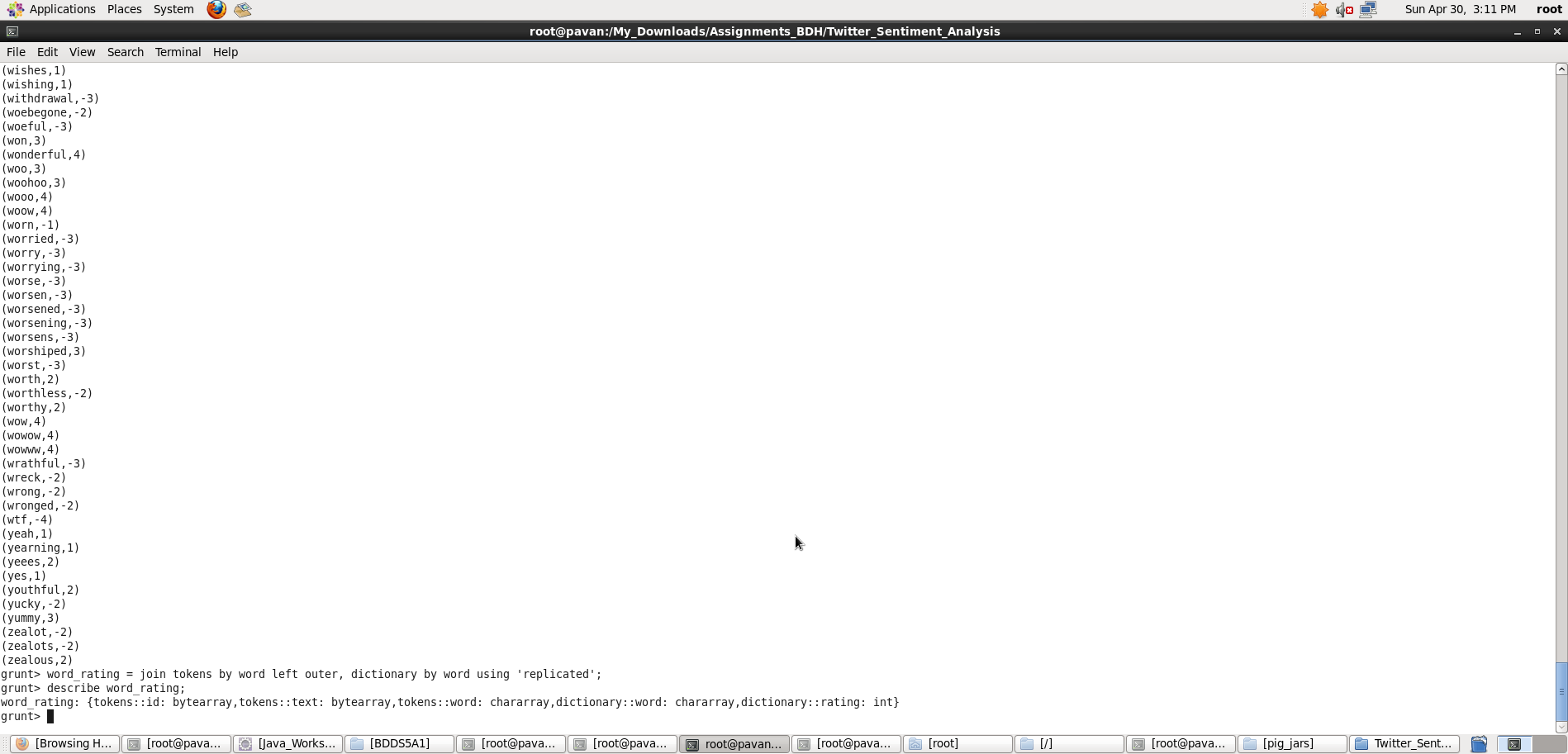
We can see the contents of the AFINN dictionary in the below screen shot.



* Now, let’s perform a map side join by joining the **tokens**statement and the dictionary contents using this command

word\_rating = join tokens by word left outer, dictionary by word using 'replicated';

* We can see the schema of the statement after performing join operation by using the describe command.

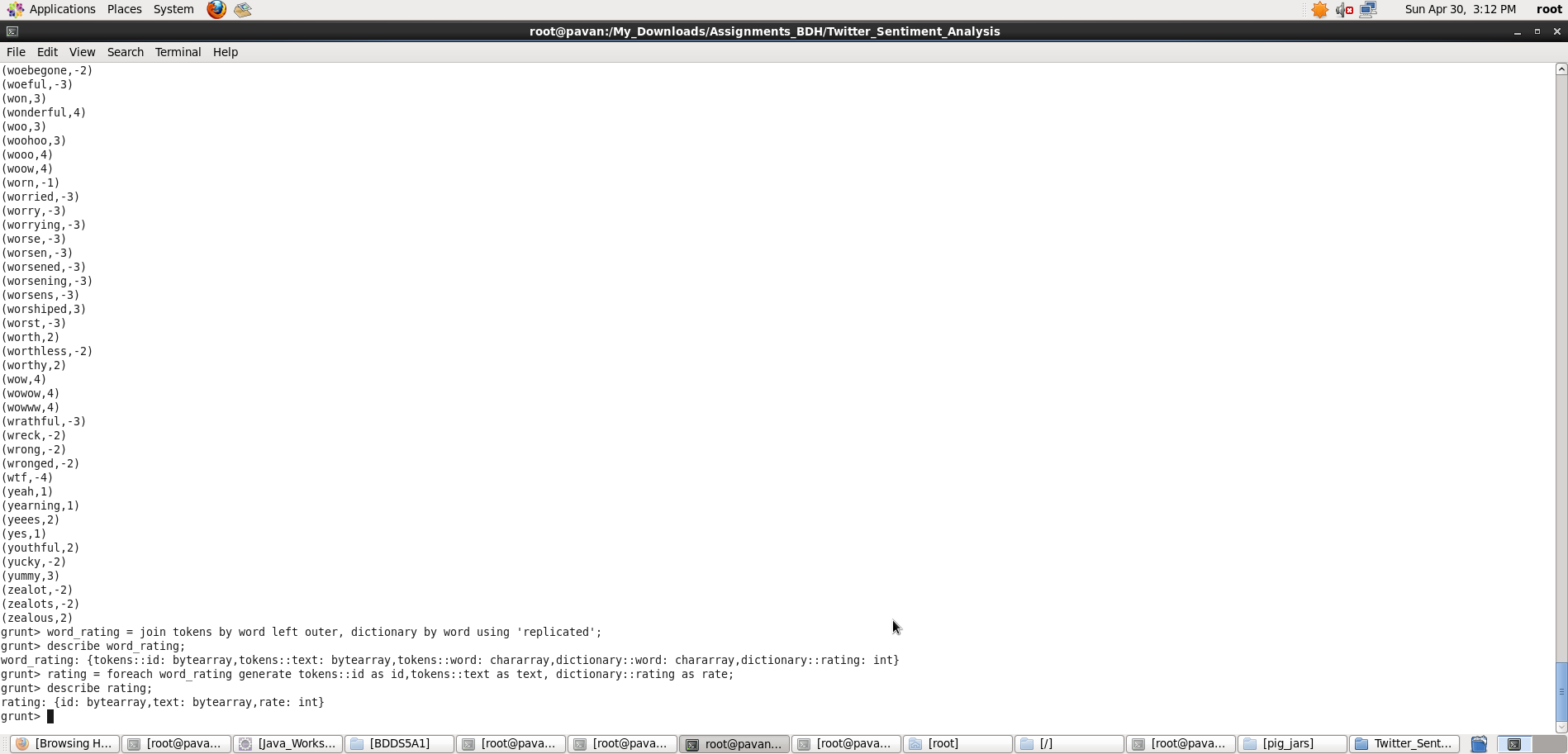


In the above screenshot, we can see that the word\_rating has joined the **tokens(**consists of id, tweet text, word**)**statement and the **dictionary**(consists of word, rating).

* Now we will extract the **id,tweet text**and **word rating(**from the dictionary**)**by using the below relation:

rating = foreach word\_rating generate tokens::id as id,tokens::text as text, dictionary::rating as rate;

We can now see the schema of the relation **rating**by using the command describe rating.



In the above screen shot we can see that our relation now consists of **id,tweet text** and **rate(**for each word**).**

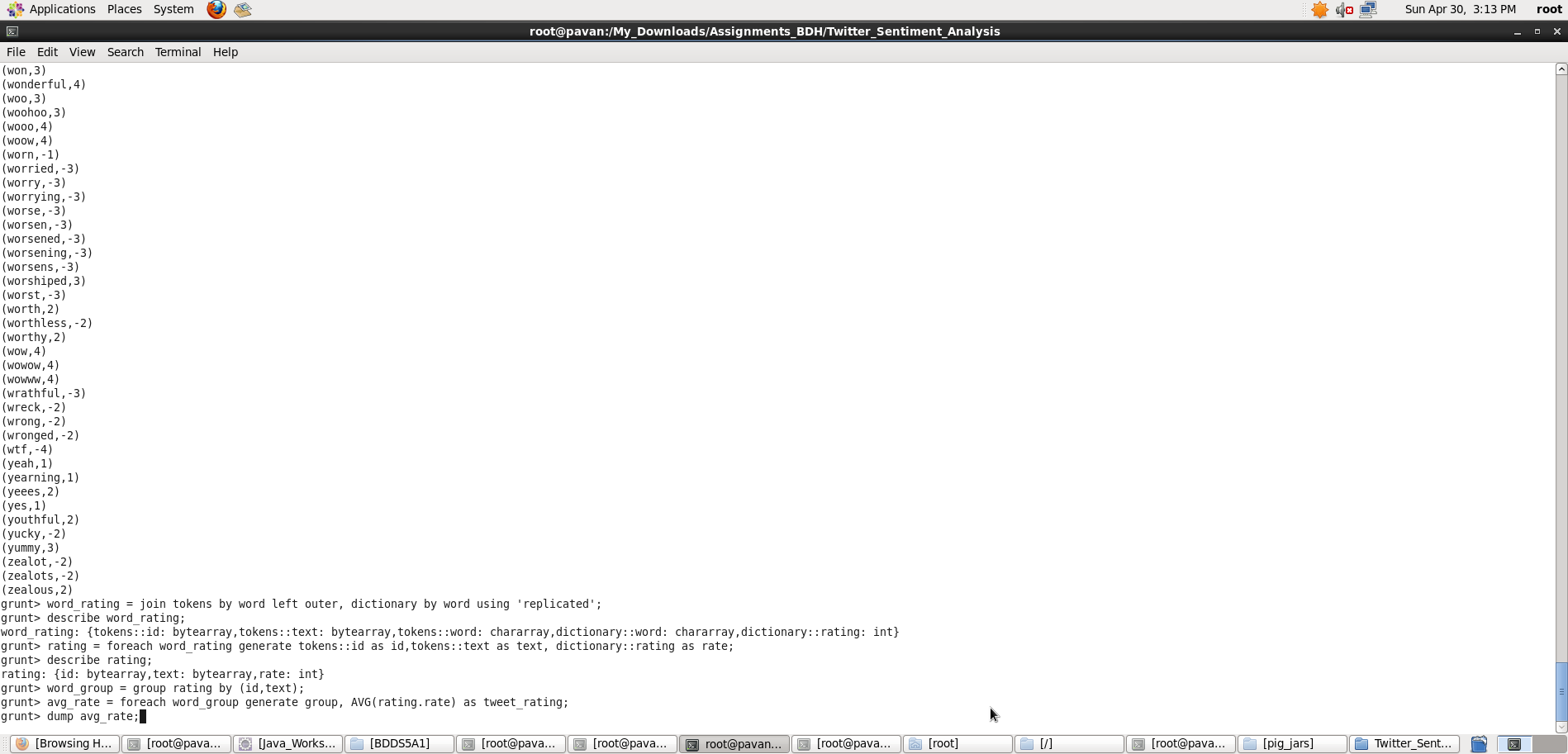
* Now, we will group the **rating of all the words in a tweet**by using the below relation:

word\_group = group rating by (id,text);

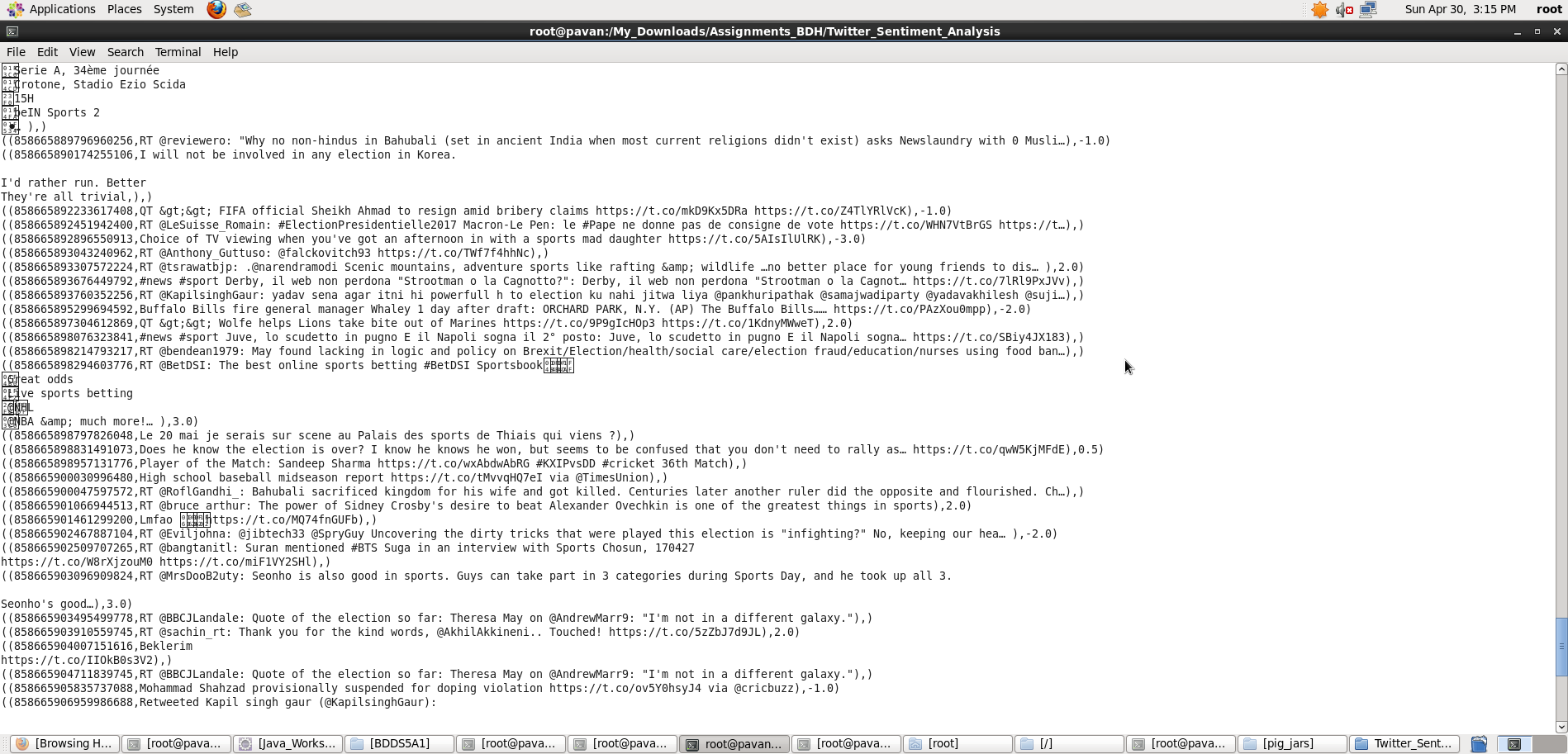
Here we have grouped by two constraints, **id** and **tweet text**.

* Now, let’s perform the **Average** operation on the **rating of the words per each tweet**.

avg\_rate = foreach word\_group generate group, AVG(rating.rate) as tweet\_rating;



**Now we have calculated the Average rating of the tweet using the rating of the each word.**You can refer to the below image for the same.



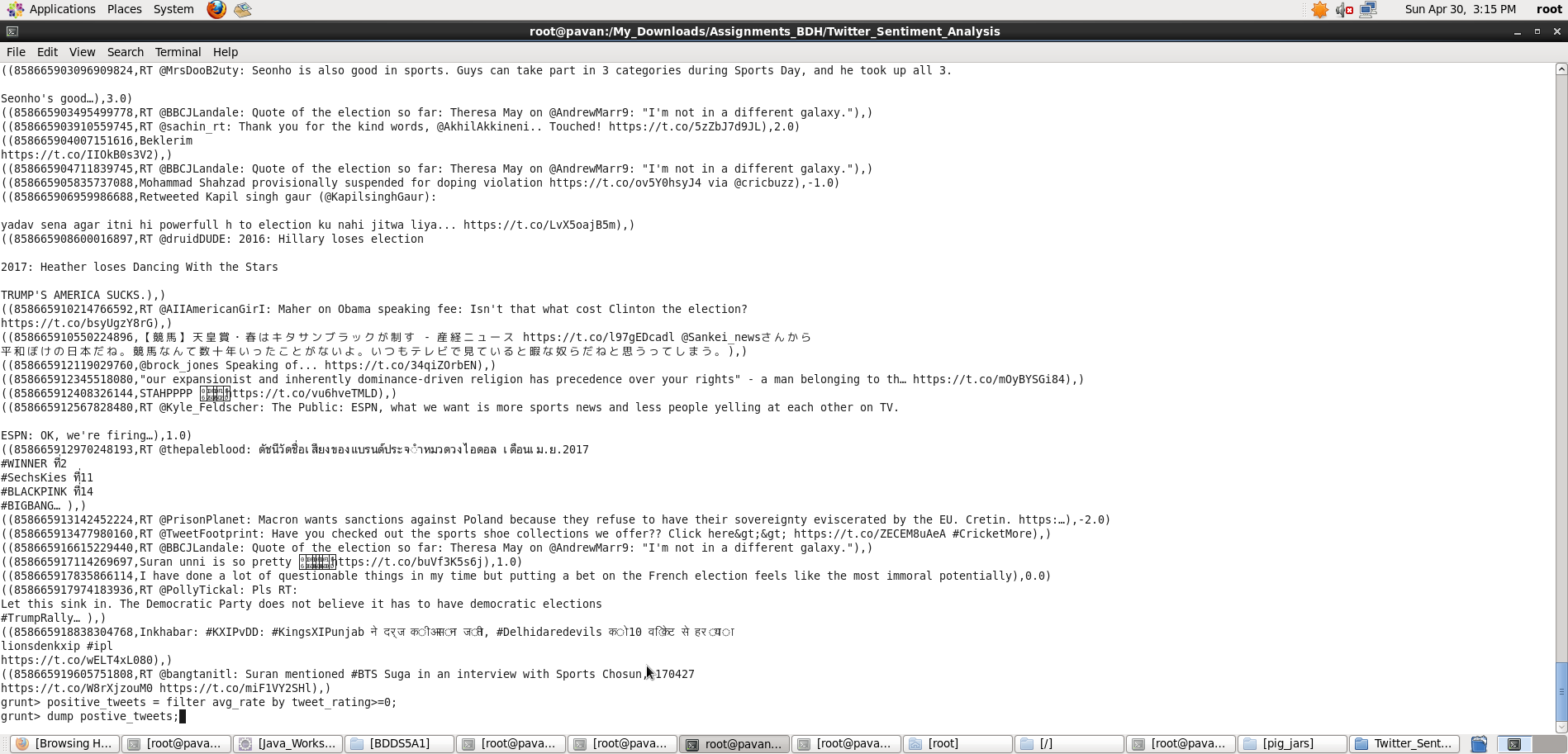
From the above relation, we will get all the tweets i.e., both positive and negative.

Here, we can classify the positive tweets by taking the rating of the tweet which can be from **0-5.**We can classify the negative tweets by taking the rating of the tweet from**-5 to -1.**

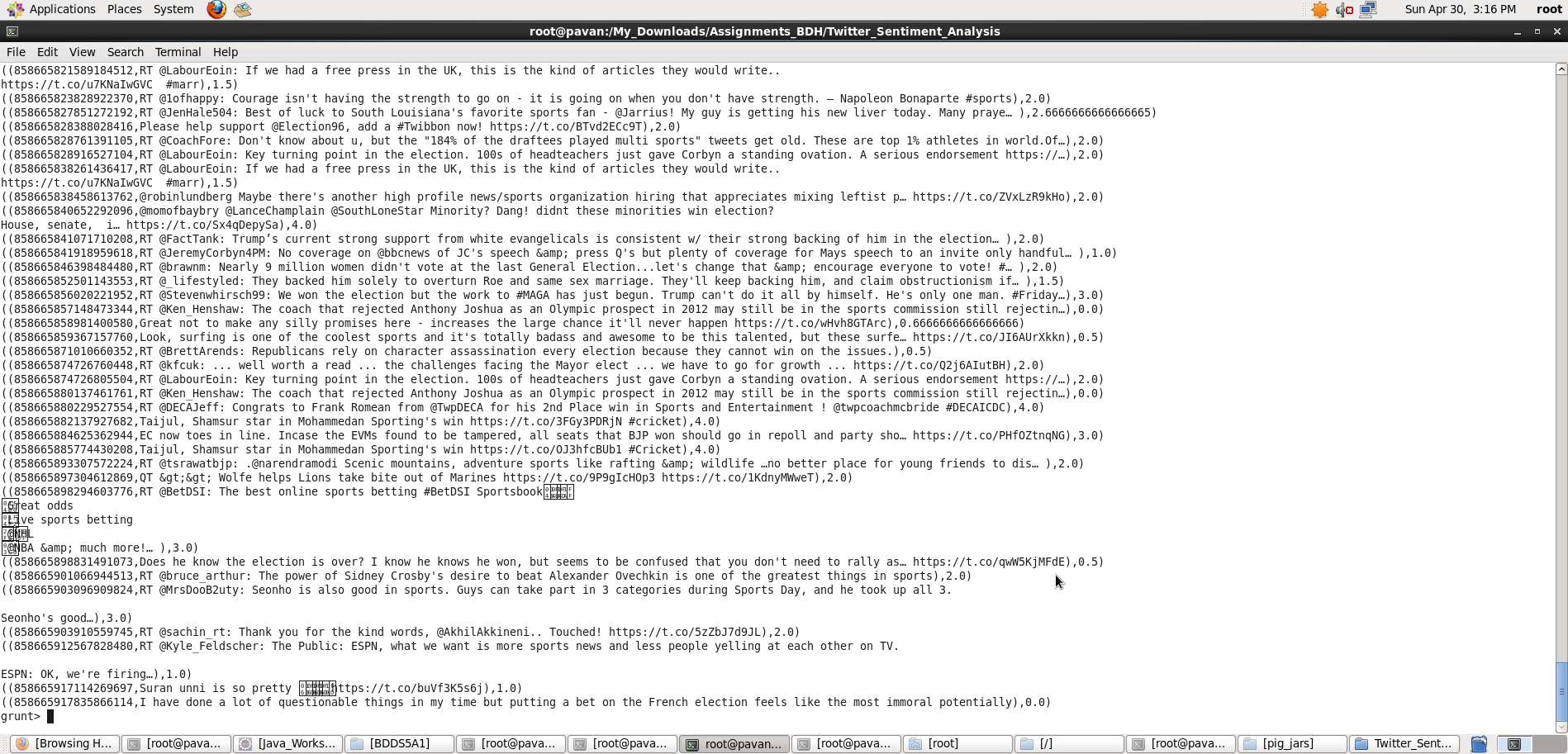
We have now successfully performed the Sentiment Analysis on Twitter data using Pig. We now have the tweets and its rating, so let’s perform an operation to filter out the positive tweets.

* Now we will filter the positive tweets using the below statement:

positive\_tweets = filter avg\_rate by tweet\_rating>=0;



We can see the positive tweets and its rating in the below screen shot.



In the above screen shot we can see the tweet\_id,tweet\_text and its rating.

**Thus, we performed Twitter Sentiment Analysis using Pig commands in Grunt shell by following the Acadgild Twitter Sentiment Analysis Blog.**