Spark Assignment 22.2

This assignment is about **Sentiment analysis on demonetization**

Objective: Let us find out the views of different people on the demonetization by analysing the tweets from twitter. Here is the dataset where twitter tweets are gathered in CSV format. The data set is given in the below link. So download the dataset from the below link

https://drive.google.com/open?id=0ByJLBTmJojjzNkRsZWJiY1VGc28

The dataset contains 2 files, one is csv file, which contains tweet related details. Another is a .txt file, which contains words and rating.

This assignment is done in the spark shell of Acadgild VM.

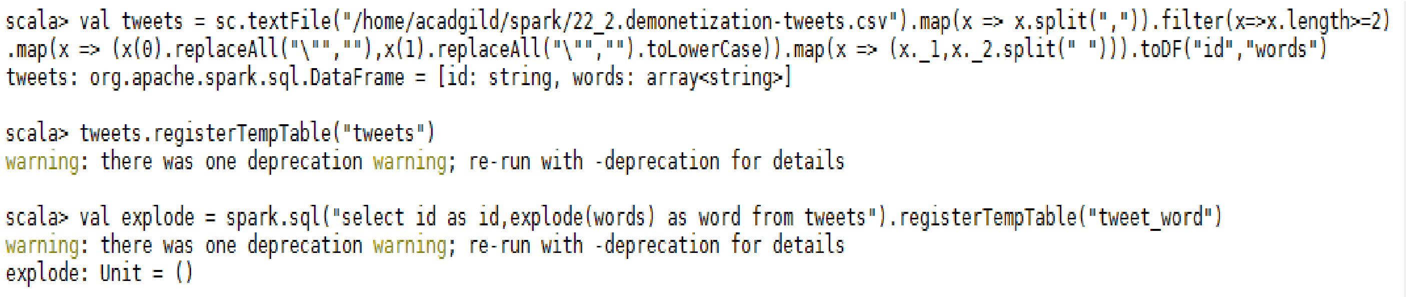
Steps Followed:

1) Copied the dataset file (CSV) in the path /home/acadgild/22_2.demonetization-tweets.csv. Then read the text file by using sc.textfile, then Splitting the words by “,”. If the line has more than 2 items, then replacing the special symbols and converting the word into lower case as below. Finally, making it as Dataframe and registering it as a temporary table “tweets”

```
val tweets = sc.textFile("/home/acadgild/spark/22_2.demonetization-tweets.csv").map(x =>
x.split(",")).filter(x=>x.length>=2).map(x =>
(x(0).replaceAll("\"",""),x(1).replaceAll("\"","").toLowerCase)).map(x => (x._1,x._2.split("
"))).toDF("id","words")
tweets.registerTempTable("tweets")
```

2) Selecting id and words from tweets table. Using explode function over words of tweets table and return a new Dataset where a single column has been expanded to zero or more rows by the provided function. Finally, registering it as a temporary table “tweet_words”

```
val explode = spark.sql("select id as id,explode(words) as word from
tweets").registerTempTable("tweet_word")
```



```
scala> val tweets = sc.textFile("/home/acadgild/spark/22_2.demonetization-tweets.csv").map(x => x.split(",")).filter(x=>x.length>=2)
.map(x => (x(0).replaceAll("\"",""),x(1).replaceAll("\"","").toLowerCase)).map(x => (x._1,x._2.split(" "))).toDF("id","words")
tweets: org.apache.spark.sql.DataFrame = [id: string, words: array<string>]

scala> tweets.registerTempTable("tweets")
warning: there was one deprecation warning; re-run with -deprecation for details

scala> val explode = spark.sql("select id as id,explode(words) as word from tweets").registerTempTable("tweet_word")
warning: there was one deprecation warning; re-run with -deprecation for details
explode: Unit = ()
```

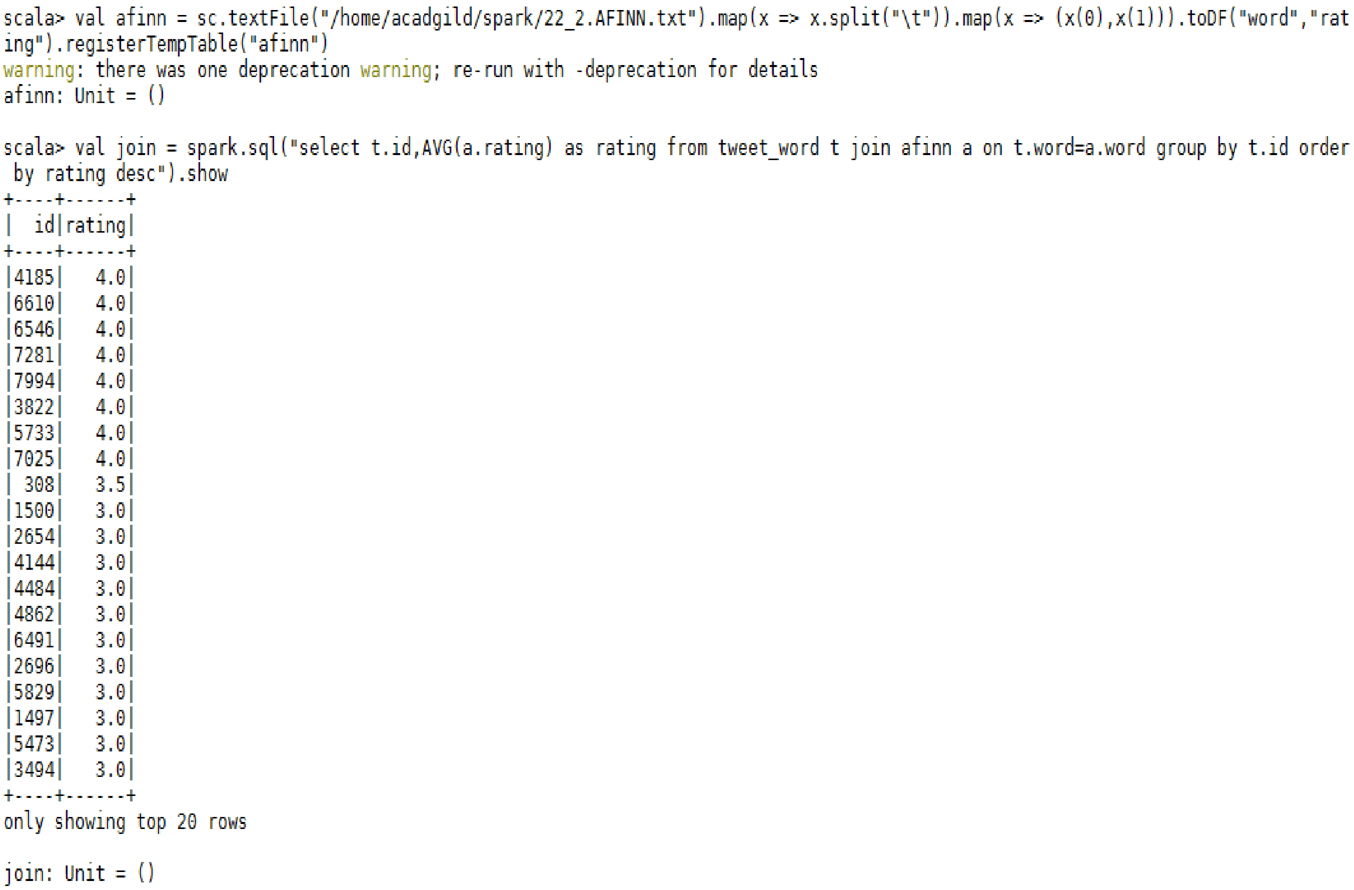
3) Copied the dataset file (txt) in the path /home/acadgild/22_2.AFINN.txt. Then read the text file by using sc.textfile, then Splitting the words by “\t”. Mapping the 2 items as word and rating. Finally, making it as Dataframe and registering it as a temporary table “afinn”

```
val afinn = sc.textFile("/home/acadgild/spark/22_2.AFINN.txt").map(x => x.split("\t")).map(x
=> (x(0),x(1))).toDF("word","rating").registerTempTable("afinn")
```

4) Finally joining the two tables over the words, finally selecting the id of the tweet and average of rating and finding the rating for each tweet id from tweets table.

```
val join = spark.sql("select t.id,AVG(a.rating) as rating from tweet_word t join afinn a on
t.word=a.word group by t.id order by rating desc").show
```

Spark Shell Output:



```
scala> val afinn = sc.textFile("/home/acadgild/spark/22_2.AFINN.txt").map(x => x.split("\t")).map(x => (x(0),x(1))).toDF("word","rat
ing").registerTempTable("afinn")
warning: there was one deprecation warning; re-run with -deprecation for details
afinn: Unit = ()

scala> val join = spark.sql("select t.id,AVG(a.rating) as rating from tweet_word t join afinn a on t.word=a.word group by t.id order
 by rating desc").show
+----+------+
|  id|rating|
+----+------+
|4185|   4.0|
|6610|   4.0|
|6546|   4.0|
|7281|   4.0|
|7994|   4.0|
|3822|   4.0|
|5733|   4.0|
|7025|   4.0|
| 308|   3.5|
|1500|   3.0|
|2654|   3.0|
|4144|   3.0|
|4484|   3.0|
|4862|   3.0|
|6491|   3.0|
|2696|   3.0|
|5829|   3.0|
|1497|   3.0|
|5473|   3.0|
|3494|   3.0|
+----+------+
only showing top 20 rows

join: Unit = ()
```

Output:

```
+----+------+
|  id|rating|
+----+------+
|4185|   4.0|
|6610|   4.0|
|6546|   4.0|
|7281|   4.0|
|7994|   4.0|
|3822|   4.0|
|5733|   4.0|
|7025|   4.0|
| 308|   3.5|
|1500|   3.0|
|2654|   3.0|
|4144|   3.0|
|4484|   3.0|
|4862|   3.0|
|6491|   3.0|
|2696|   3.0|
|5829|   3.0|
|1497|   3.0|
|5473|   3.0|
|3494|   3.0|
+----+------+
```