ASSIGNMENT 22.2

Problem Statement:

Go through below blog and reiterate the same at your end.

https://docs.google.com/document/d/14YUd_wi-KJTBEqqtyoNtPMFeORjFU-_yId4hJ11q90o/

Solution:

Sentiment analysis on demonetization:

Let us find out the views of different people on the demonetization by analyzing the tweets from twitter. Here is the dataset where twitter tweets are gathered in CSV format. You can download the dataset from the below link

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

Metadata of the tweets are as follows:

- id
- Text (Tweets)
- favorited
- favoriteCount
- replyToSN
- created
- truncated
- replyToSID
- tweetId
- replyToUID
- statusSource
- screenName
- retweetCount
- isRetweet
- retweeted

Now we will load the data into Spark as follows:

```
val tweets = sc.textFile("E:\\Acadgild\\Session 22\\Assignment 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")
```

We have defined the schema in the above line of code taking only the tweet id and the tweet from input dataset. We have got the data frame ready now. Let's register this as a temporary table to perform relevant queries down the line.

```
tweets.registerTempTable("tweets")
```

Now let's split the words in each tweet by space and store them onto another table.

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

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. You can download the dictionary from the following link:
AFINN dictionary

Metadata of the dictionary file is as follows:

- word
- rating

We will load the dictionary into Spark by using the below statement:

```
val afinn = sc.textFile("E:\\Acadgild\\Session 22\\Assignment 22.2\\AFINN.txt")
                .map(x => x.split("\t"))
                .map(x => (x(0), x(1)))
                .toDF("word", "rating")
```

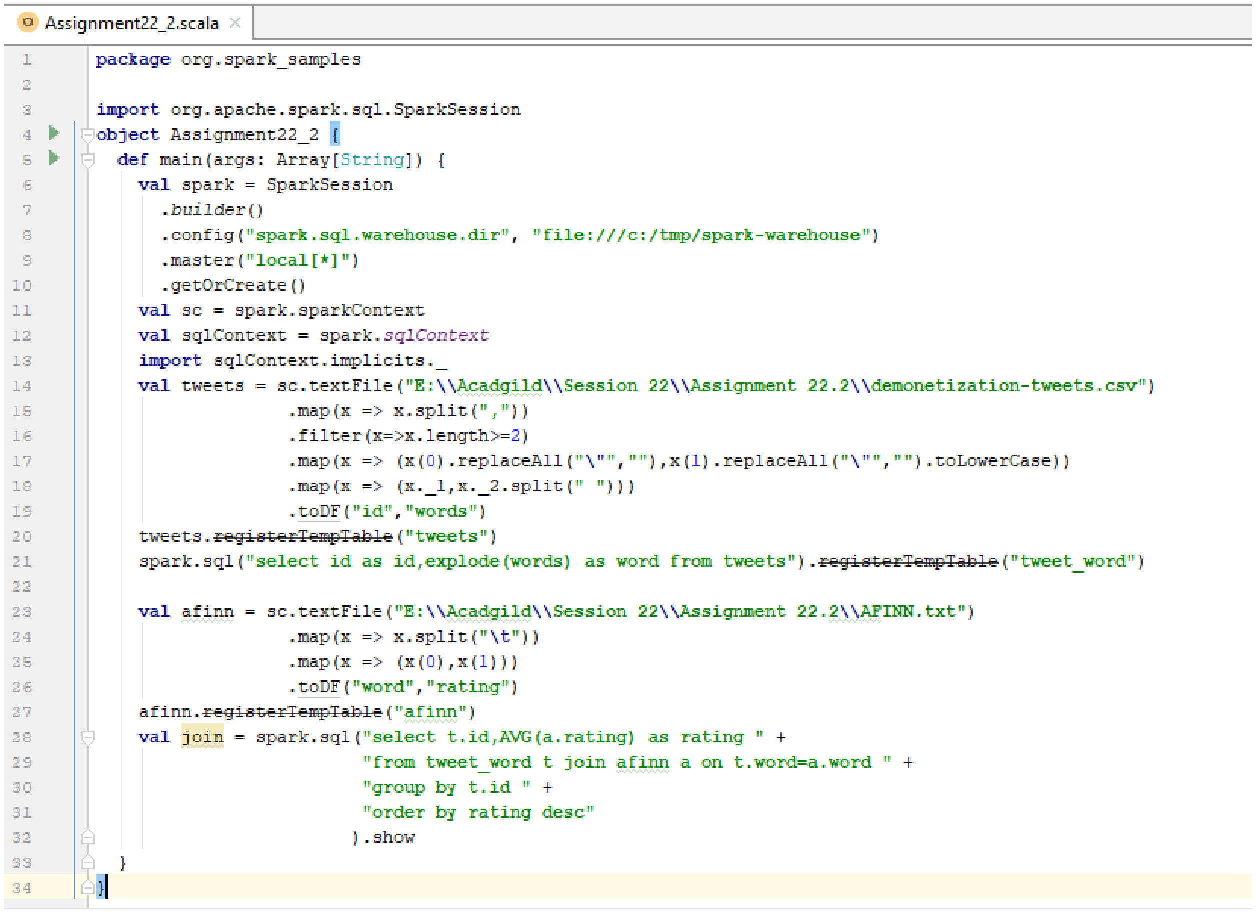
Let’s register this as a temporary table to perform analysis via SQL queries:

afinn.registerTempTable("afinn")

Now, let’s find the average rating of every word present in tweets and the dictionary by performing a map side join of the **tweet_word** and **affin tables** with matching words in both the tables which will then be grouped by tweet id and arranged in the order of rating (high to low).

```
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
```

This results in tweet id and average rating for that tweet based on the words used in that tweet and its associated rating that depicts the sentiment which can be broadly classified as positive, negative and neutral.



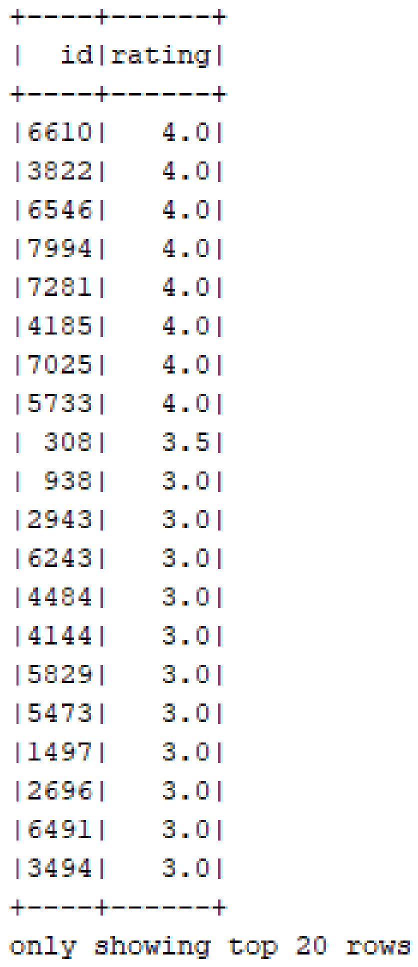
Assignment22_2.scala

```
package org.spark_samples

import org.apache.spark.sql.SparkSession
object Assignment22_2 {
  def main(args: Array[String]) {
    val spark = SparkSession
      .builder()
      .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
      .master("local[*]")
      .getOrCreate()
    val sc = spark.sparkContext
    val sqlContext = spark.sqlContext
    import sqlContext.implicits._
    val tweets = sc.textFile("E:\\Acadgild\\Session 22\\Assignment 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")
    spark.sql("select id as id,explode(words) as word from tweets").~~registerTempTable~~("tweet_word")

    val afinn = sc.textFile("E:\\Acadgild\\Session 22\\Assignment 22.2\\AFINN.txt")
                  .map(x => x.split("\t"))
                  .map(x => (x(0),x(1)))
                  .toDF("word","rating")
    afinn.~~registerTempTable~~("afinn")
    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
  }
}
```

Output:



```
+----+------+
|  id|rating|
+----+------+
|6610|   4.0|
|3822|   4.0|
|6546|   4.0|
|7994|   4.0|
|7281|   4.0|
|4185|   4.0|
|7025|   4.0|
|5733|   4.0|
| 308|   3.5|
| 938|   3.0|
|2943|   3.0|
|6243|   3.0|
|4484|   3.0|
|4144|   3.0|
|5829|   3.0|
|5473|   3.0|
|1497|   3.0|
|2696|   3.0|
|6491|   3.0|
|3494|   3.0|
+----+------+
only showing top 20 rows
```