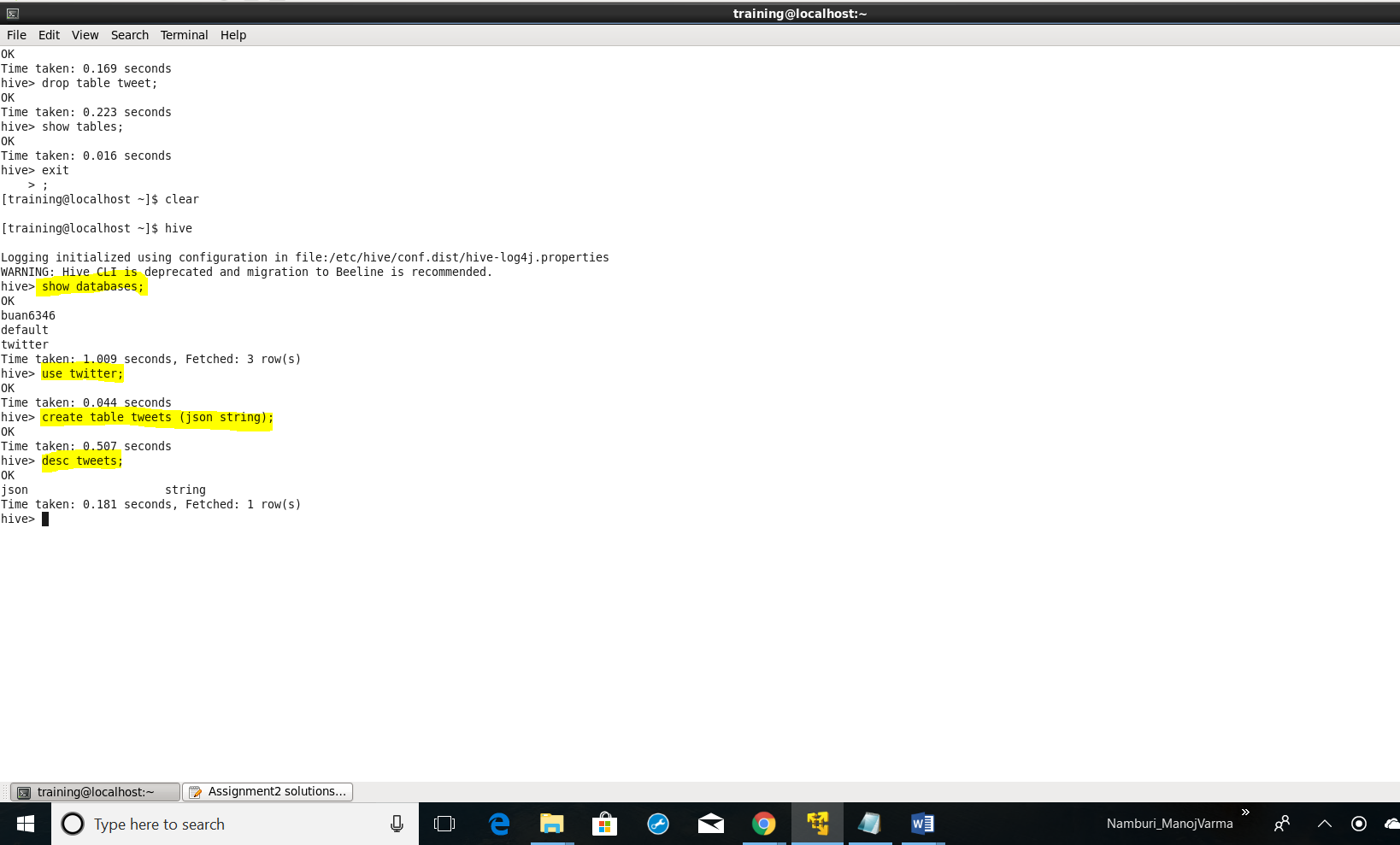
**Big Data Assignment 2**

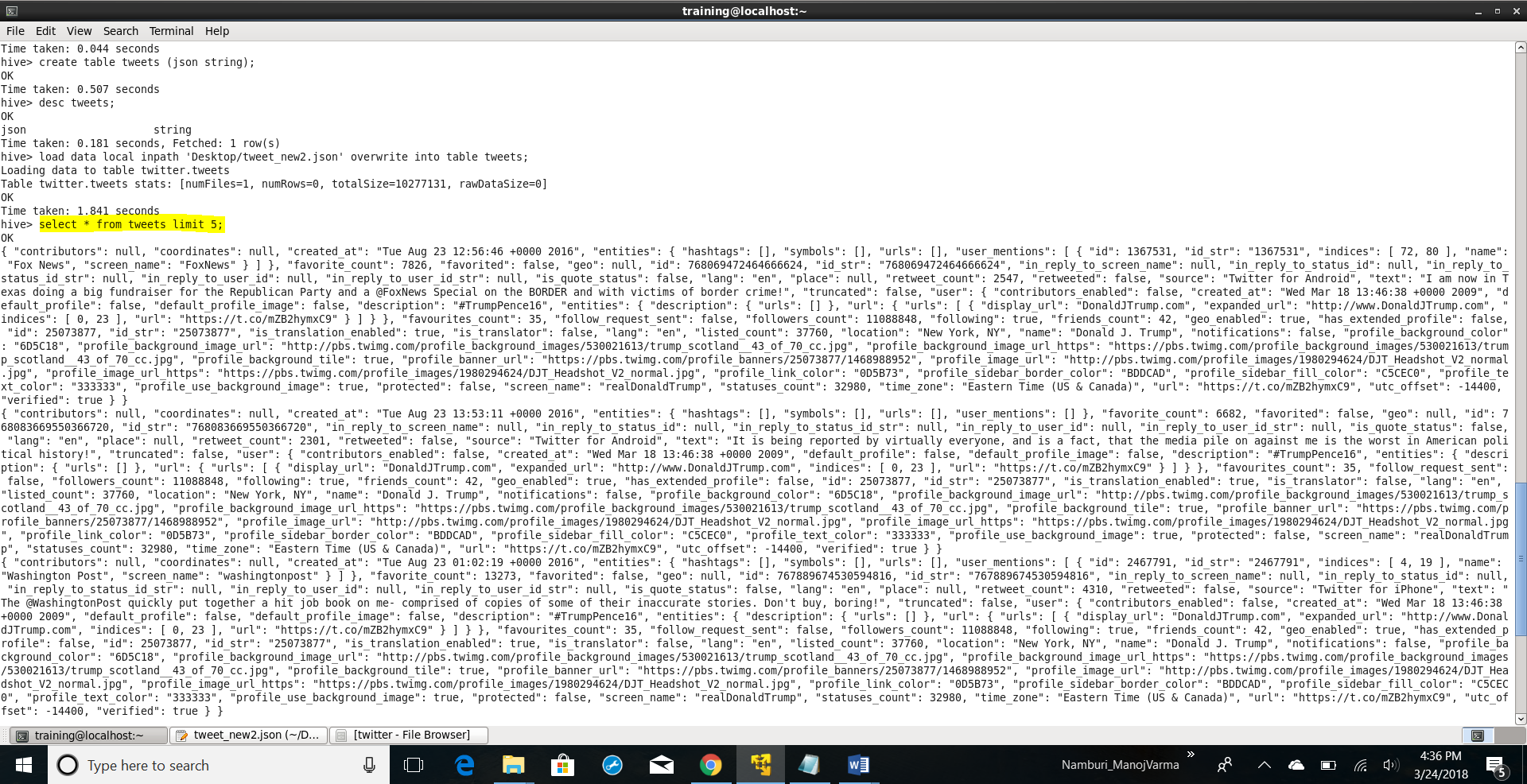
Creating database, necessary tables and loading data before answering questions

create database twitter;

Use twitter;

create table tweets (json string);



load data local inpath 'Desktop/tweet\_new2.json' overwrite into table tweets;



create table tweet\_text as

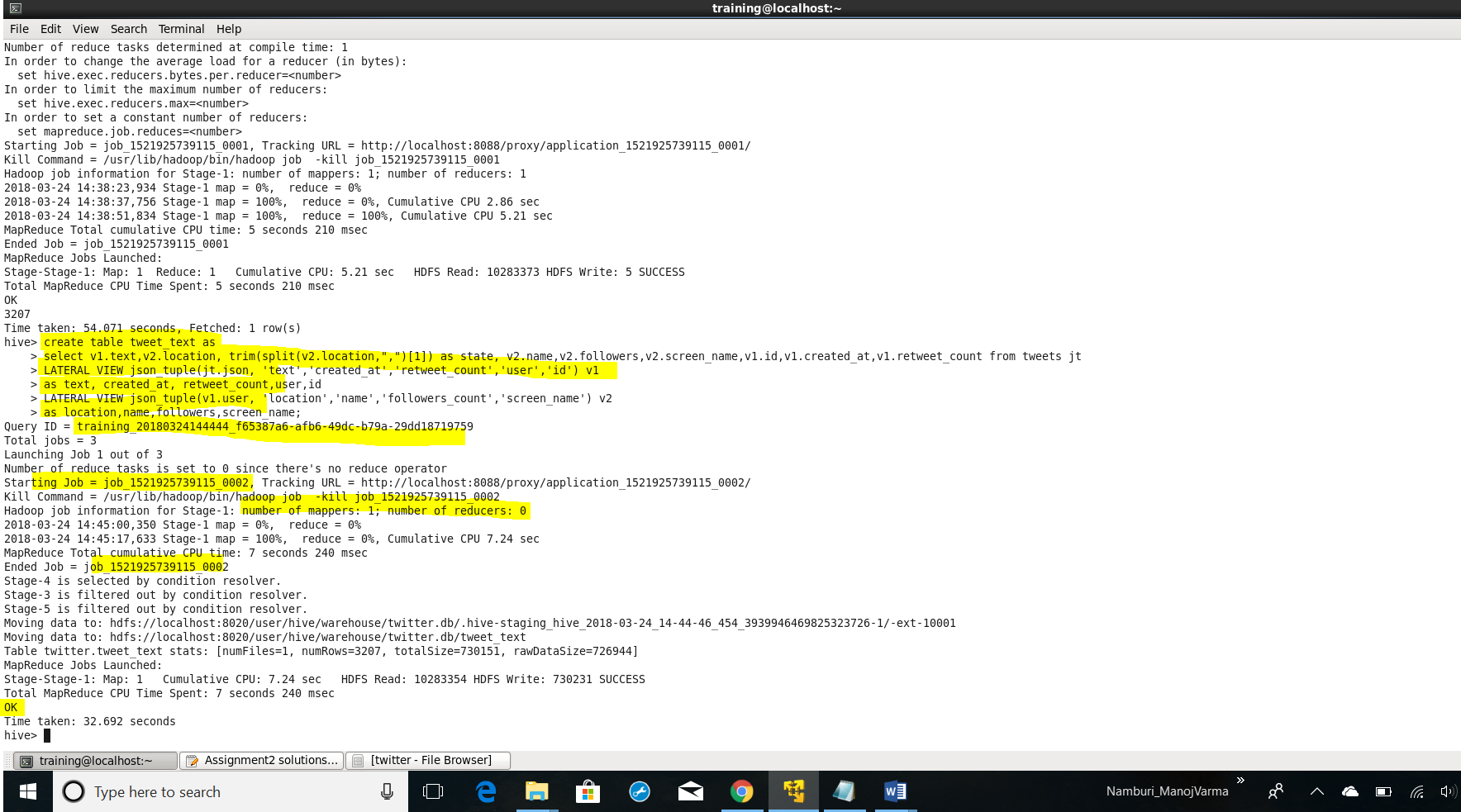
select v1.text,v2.location, trim(split(v2.location,",")[1]) as state, v2.name,v2.followers,v2.screen\_name,v1.id,v1.created\_at,v1.retweet\_count from tweets jt

LATERAL VIEW json\_tuple(jt.json, 'text','created\_at','retweet\_count','user','id') v1

as text, created\_at, retweet\_count,user,id

LATERAL VIEW json\_tuple(v1.user, 'location','name','followers\_count','screen\_name') v2

as location,name,followers,screen\_name;



**Question1:**

1. What are the hashtags used and how many times each hashtag is used?

SELECT word, count(1) as wcount from tweet\_text t

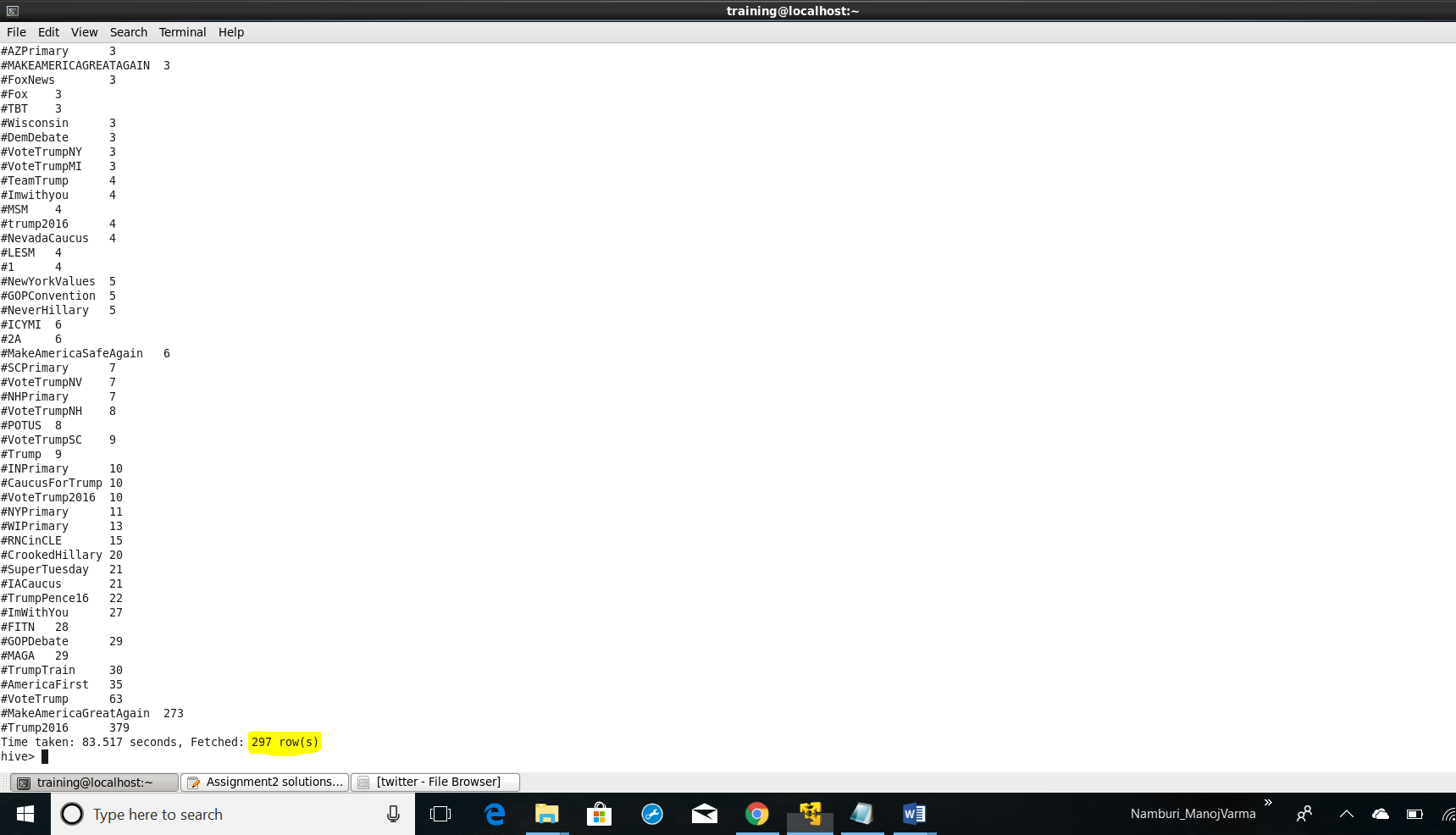
LATERAL VIEW explode(split(regexp\_replace(trim(t.text),"[^#A-Za-z0-9]"," "), ' ')) v1 as word

WHERE word rlike "^#[a-zA-Z0-9]+$"

GROUP BY word

ORDER BY wcount;



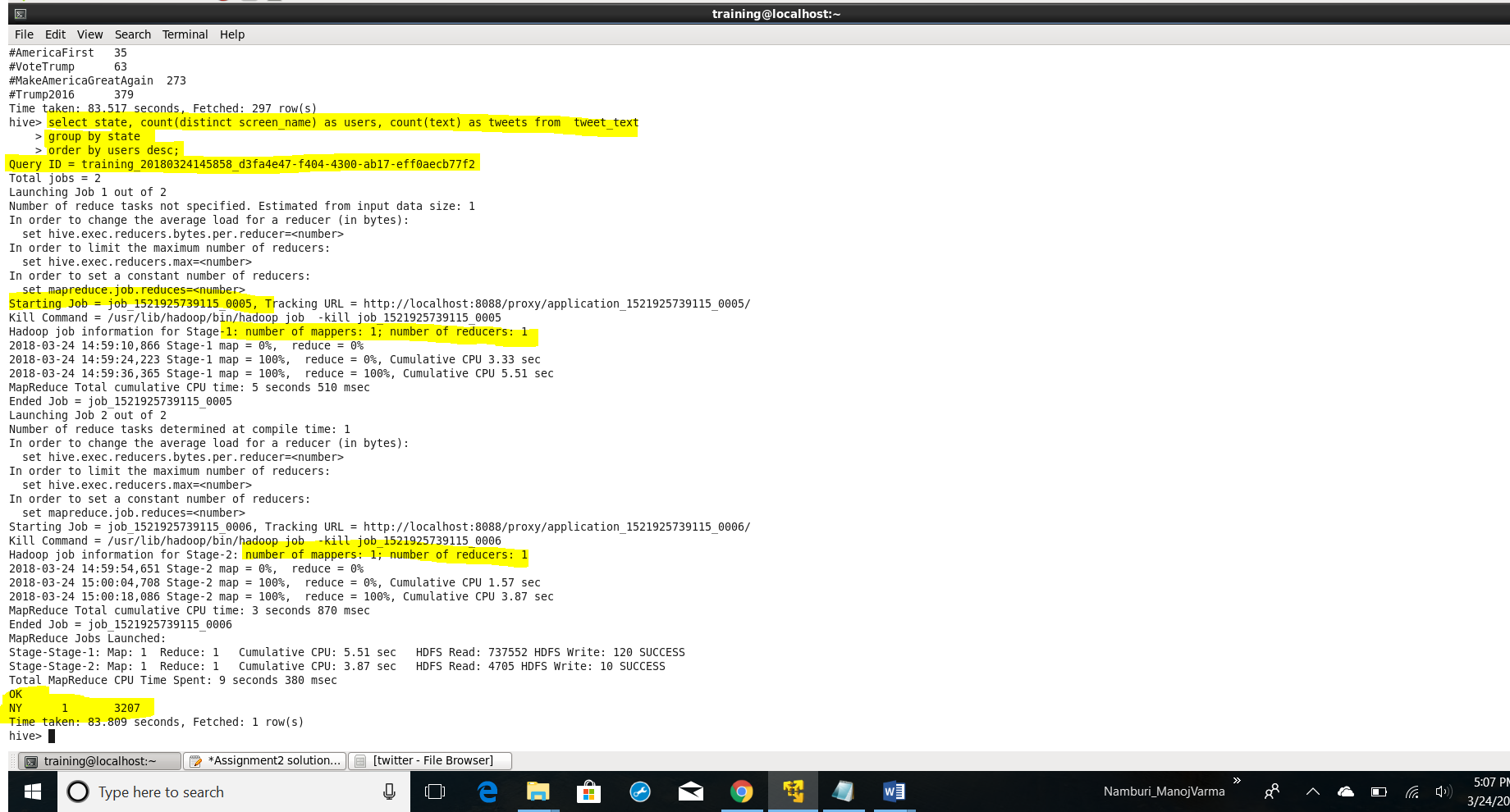


1. Which State have the most active users and how many tweets are posted by State?

select state, count(distinct screen\_name) as users, count(text) as tweets from tweet\_text

group by state

order by users desc;

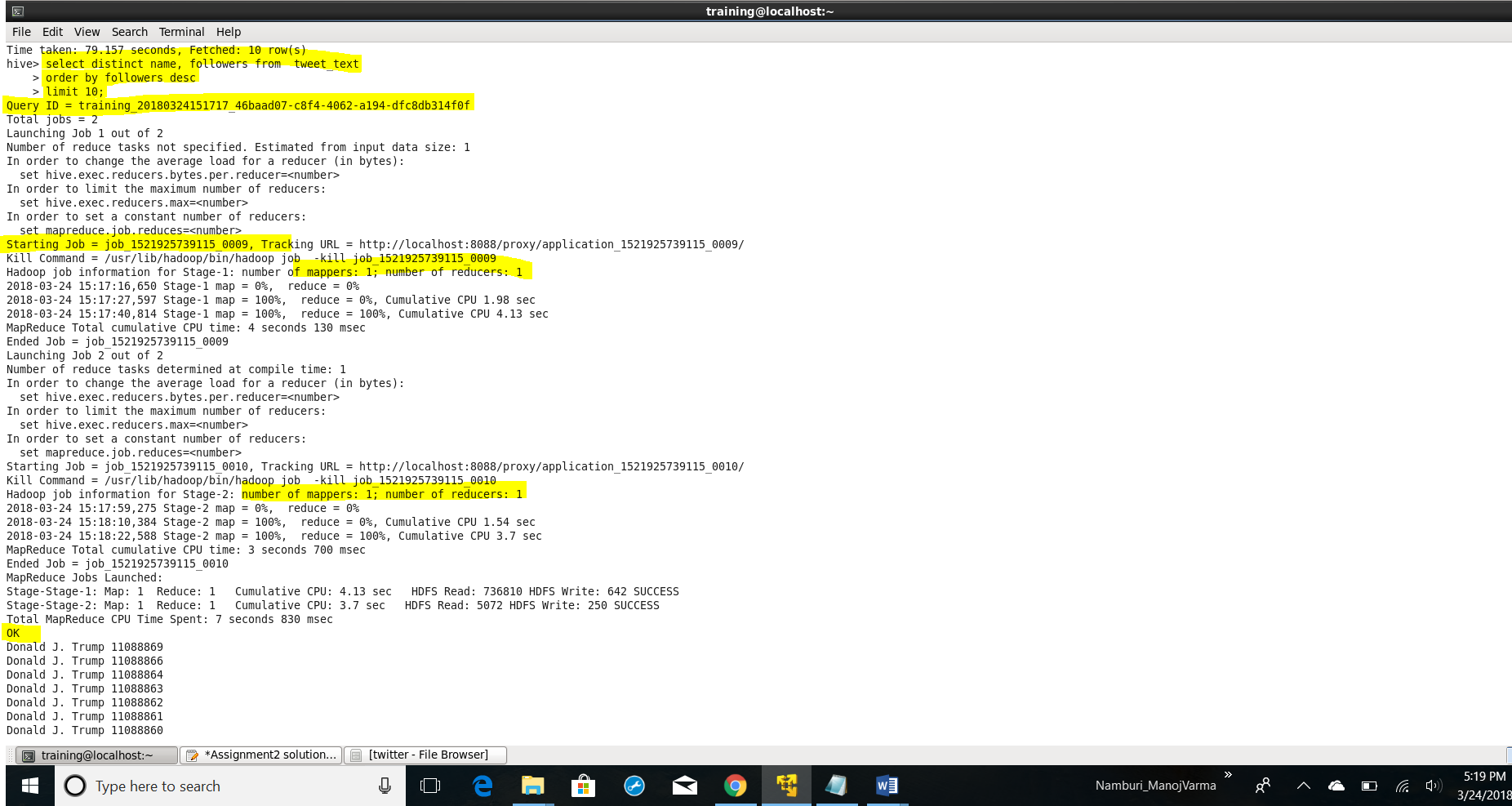


1. Based on the user’s followers count, who are the top ten users who have tweeted?

select distinct name, followers from tweet\_text

order by followers desc

limit 10;





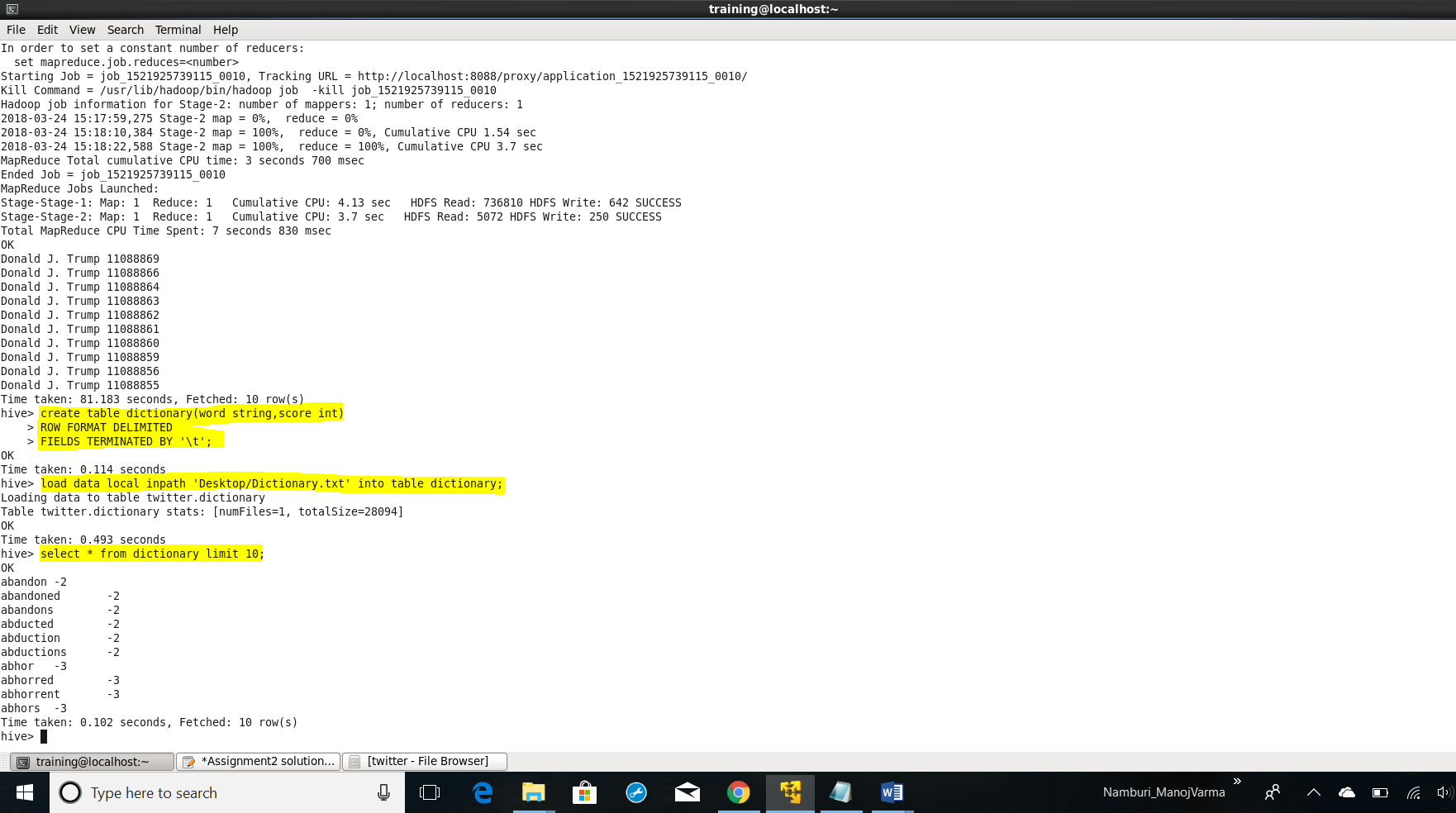
1. What is the polarity score for each tweet that was posted? Does the tweet have a positive or negative sentiment?

create table dictionary(word string,score int)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY '\t';

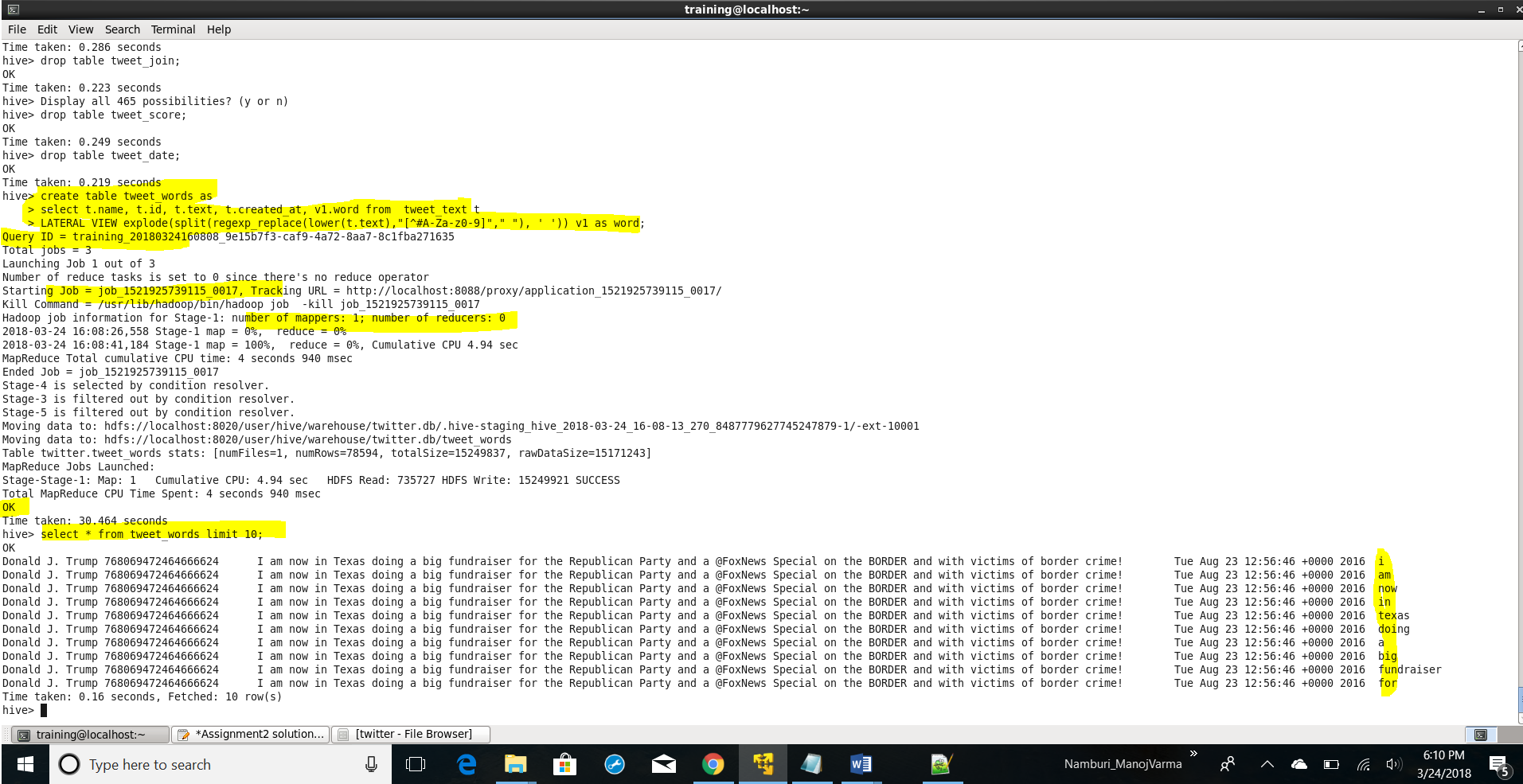
load data local inpath 'Desktop/Dictionary.txt' into table dictionary;



create table tweet\_words as

select t.name, t.id, t.text, t.created\_at, v1.word from tweet\_text t

LATERAL VIEW explode(split(regexp\_replace(lower(t.text),"[^#A-Za-z0-9]"," "), ' ')) v1 as word;



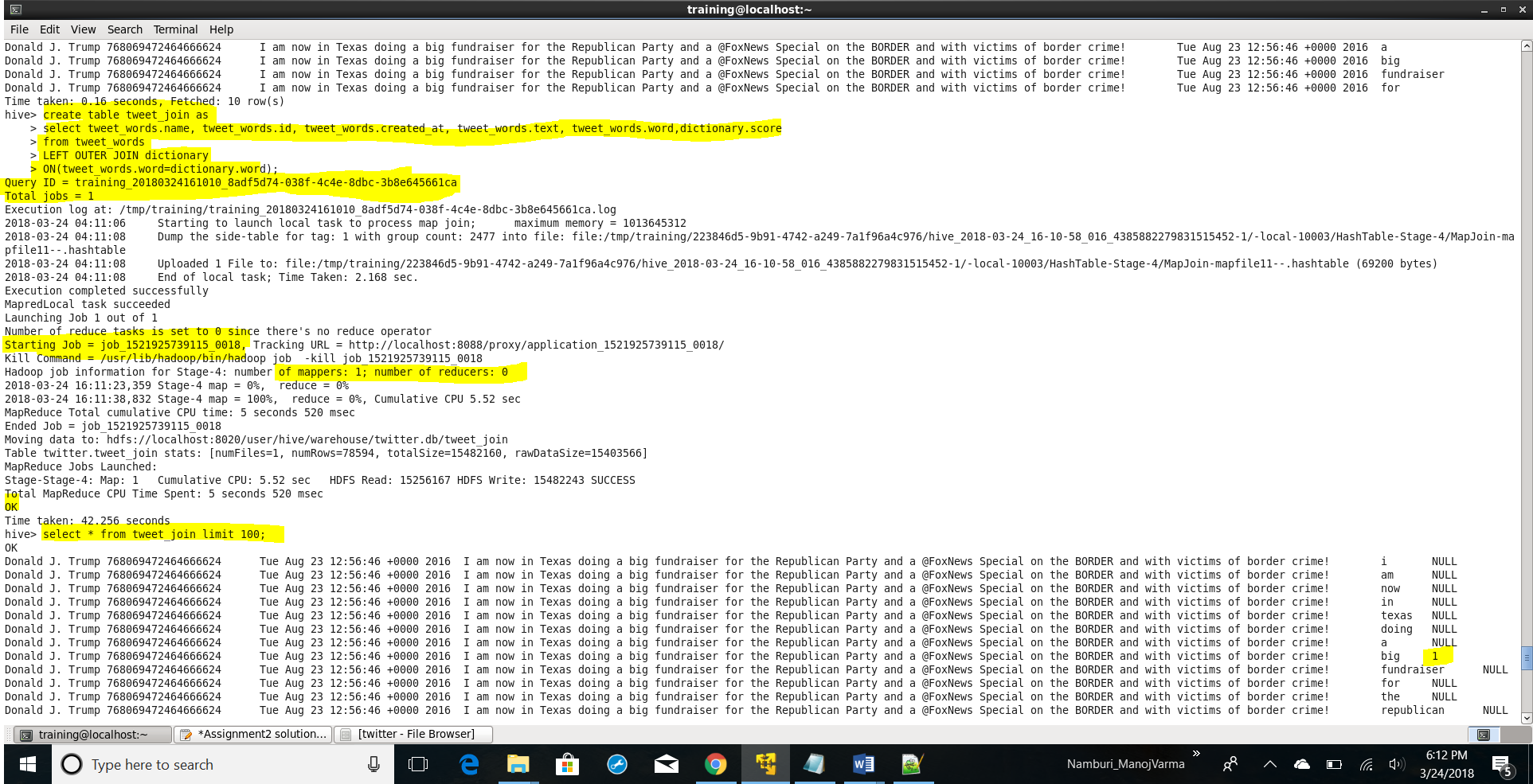
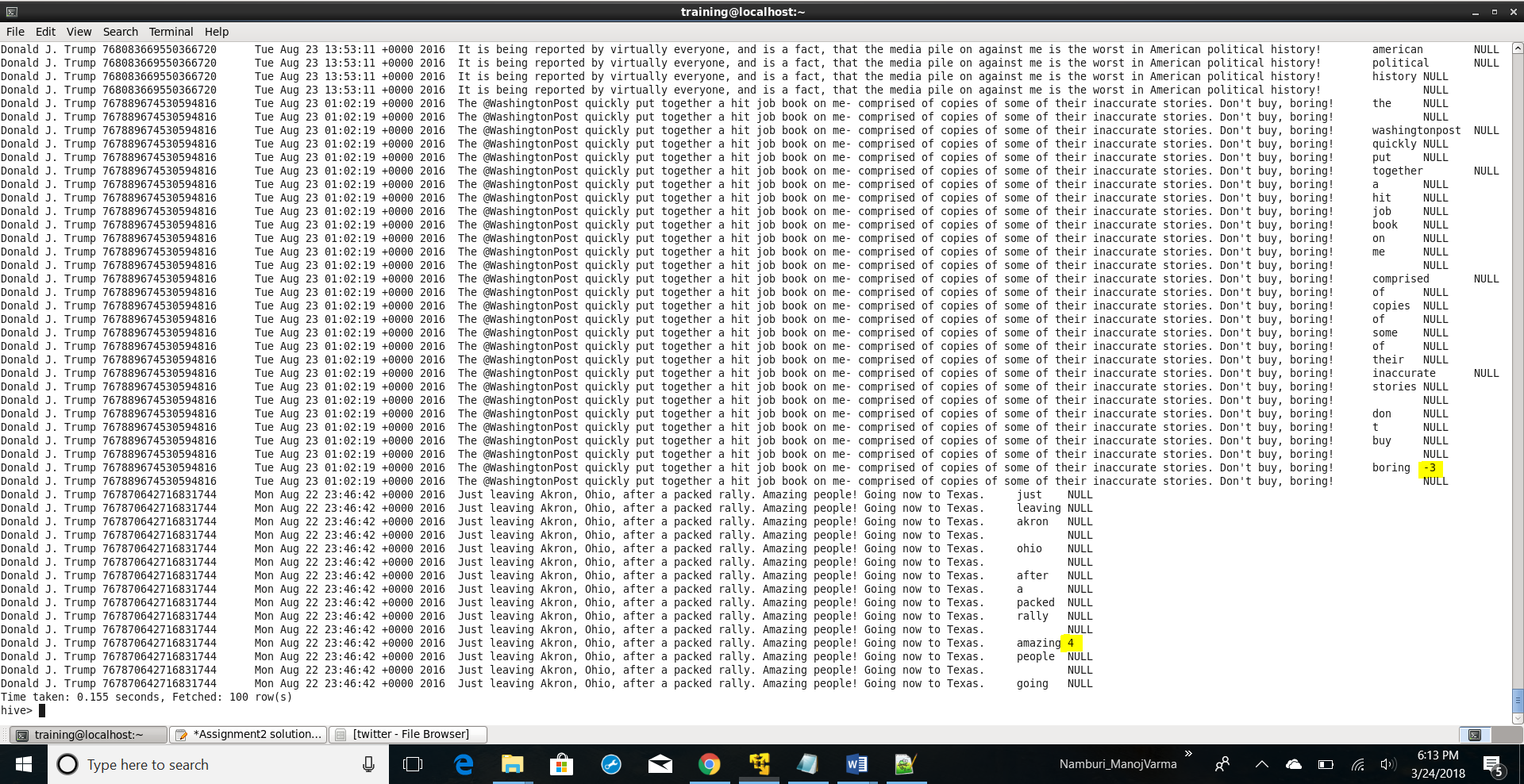
create table tweet\_join as

select tweet\_words.name, tweet\_words.id, tweet\_words.created\_at, tweet\_words.text, tweet\_words.word,dictionary.score

from tweet\_words

LEFT OUTER JOIN dictionary

ON(tweet\_words.word=dictionary.word);

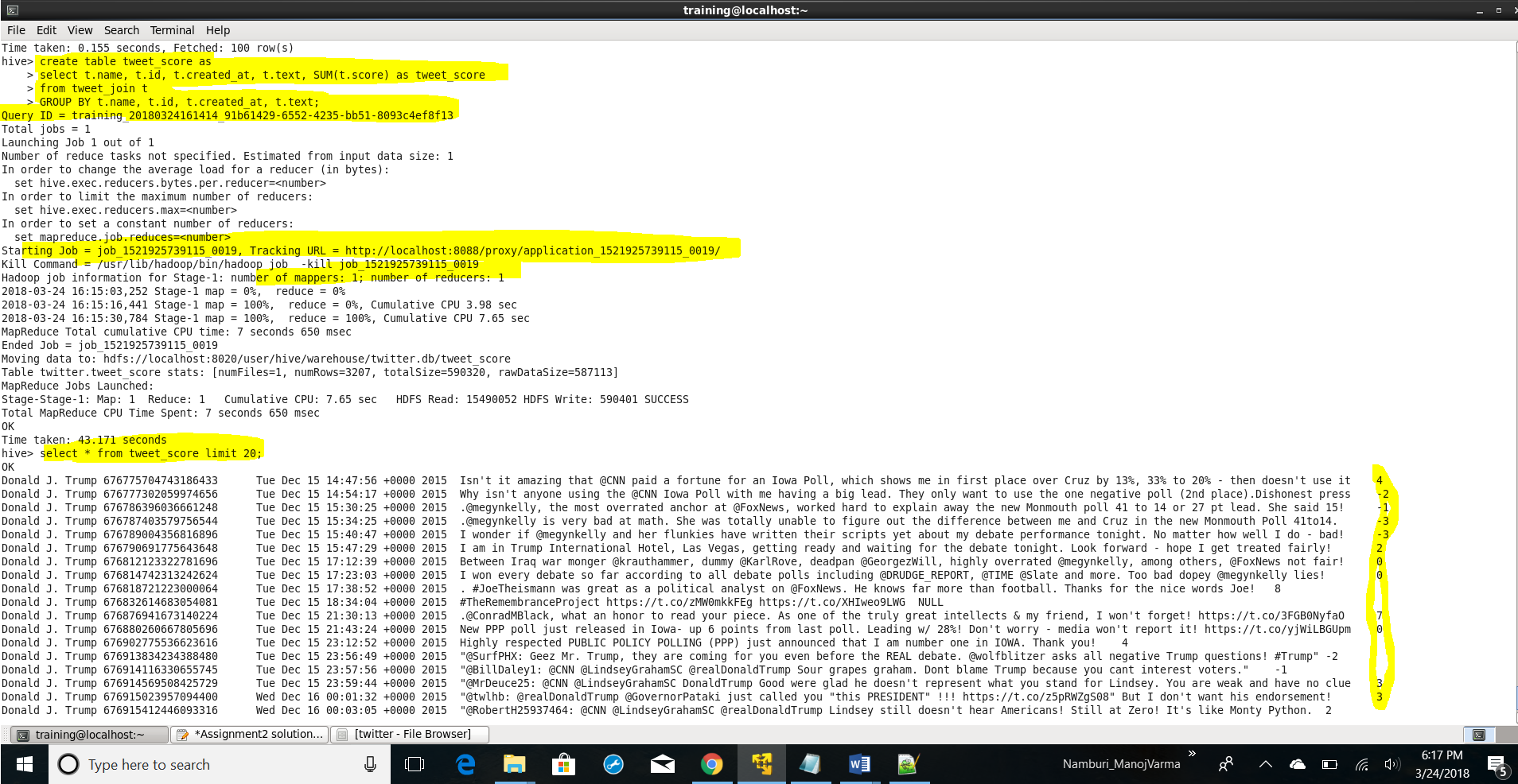
 

create table tweet\_score as

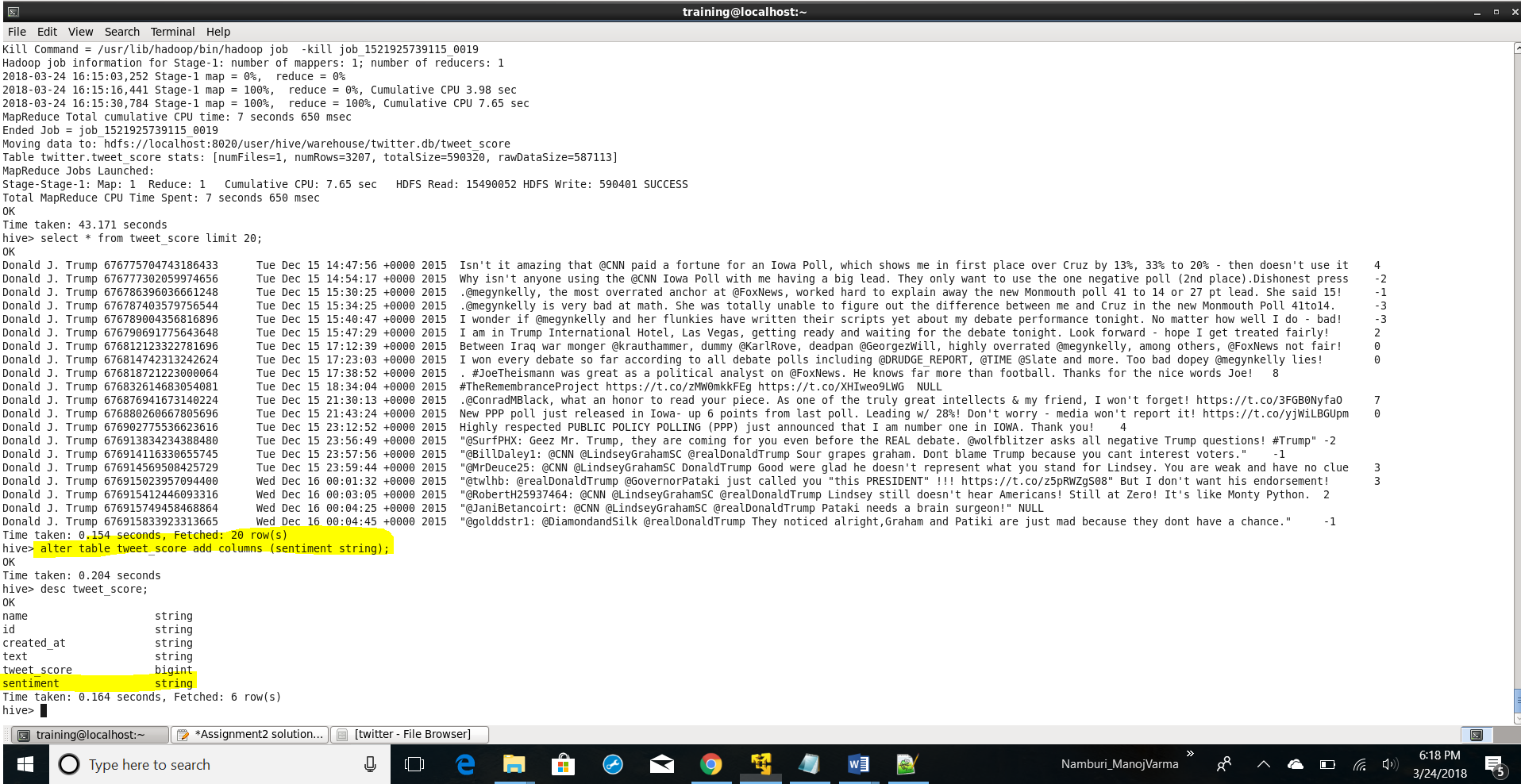
select t.name, t.id, t.created\_at, t.text, SUM(t.score) as tweet\_score

from tweet\_join t

GROUP BY t. name, t.id, t.created\_at, t.text;



alter table tweet\_score add columns (sentiment string);

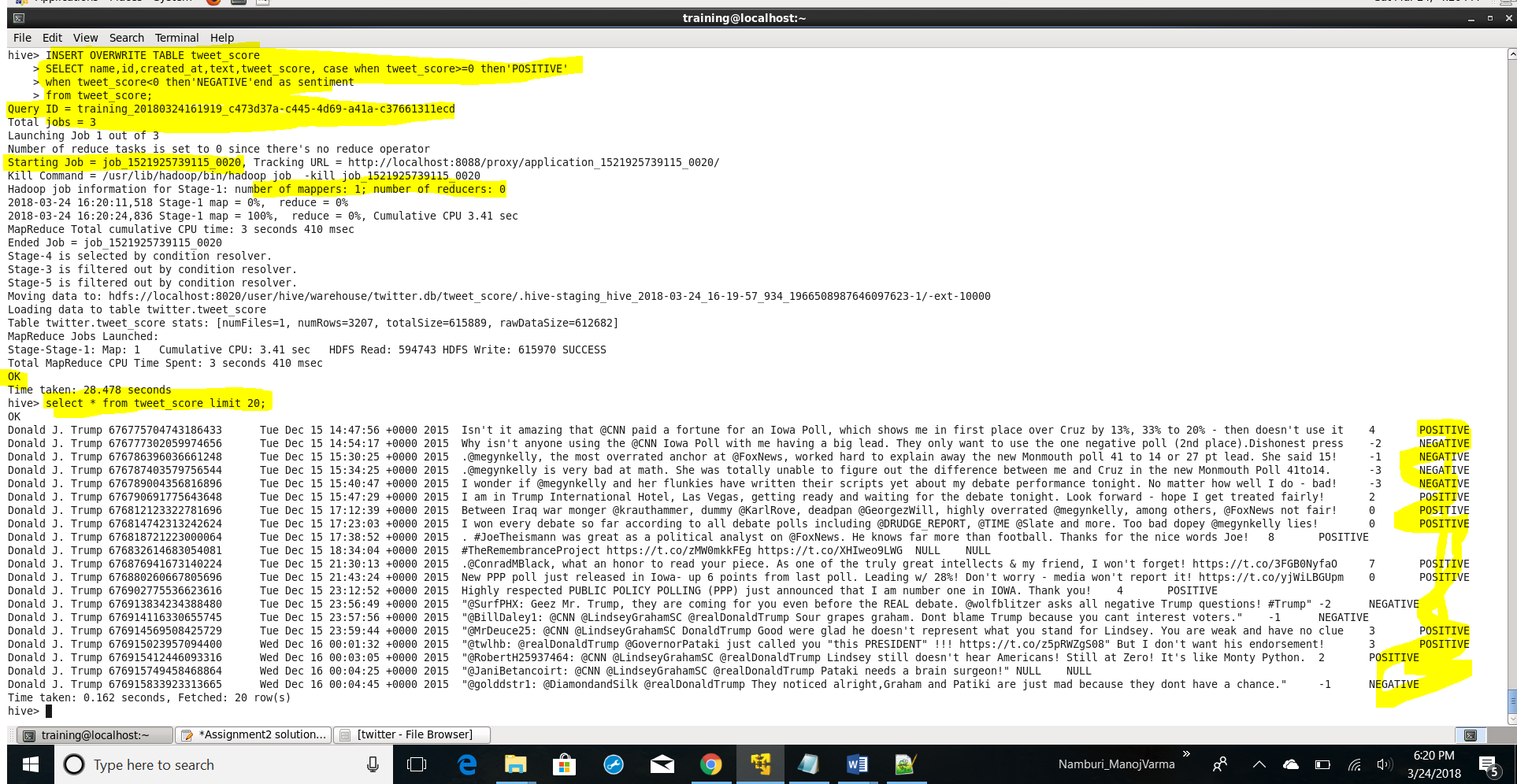


INSERT OVERWRITE TABLE tweet\_score

SELECT name,id,created\_at,text,tweet\_score, case when tweet\_score>=0 then'POSITIVE'

when tweet\_score<0 then'NEGATIVE'end as sentiment

from tweet\_score



create table tweet\_date as

select \*,

substr(created\_at,9,2) as dd, substr(created\_at,length(created\_at)-3,4) as yyyy,

case when substr(created\_at,5,3) = 'Jan' then '01'

when substr(created\_at,5,3) = 'Feb' then '02'

when substr(created\_at,5,3) = 'Mar' then '03'

when substr(created\_at,5,3) = 'Apr' then '04'

when substr(created\_at,5,3) = 'May' then '05'

when substr(created\_at,5,3) = 'Jun' then '06'

when substr(created\_at,5,3) = 'Jul' then '07'

when substr(created\_at,5,3) = 'Aug' then '08'

when substr(created\_at,5,3) = 'Sep' then '09'

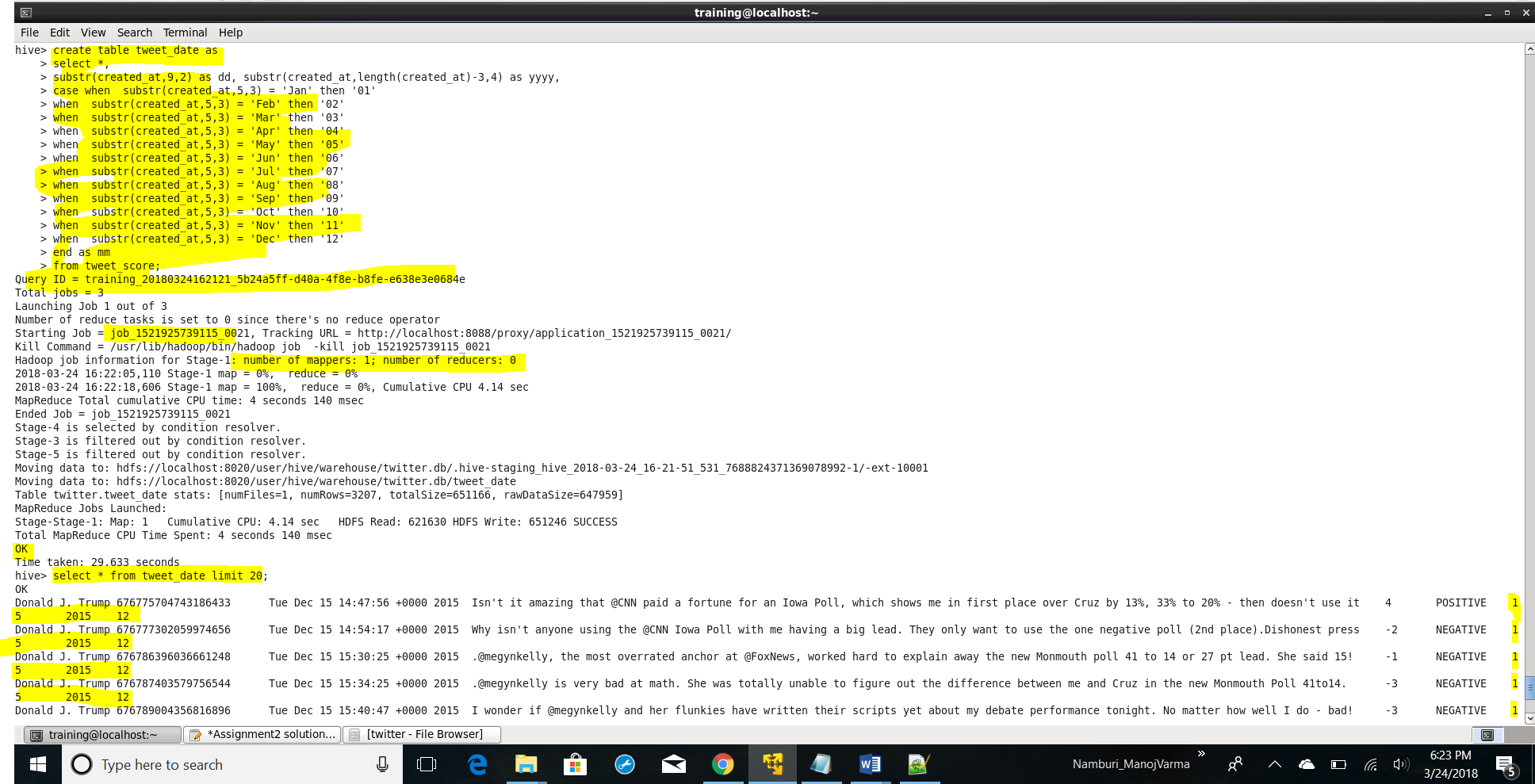
when substr(created\_at,5,3) = 'Oct' then '10'

when substr(created\_at,5,3) = 'Nov' then '11'

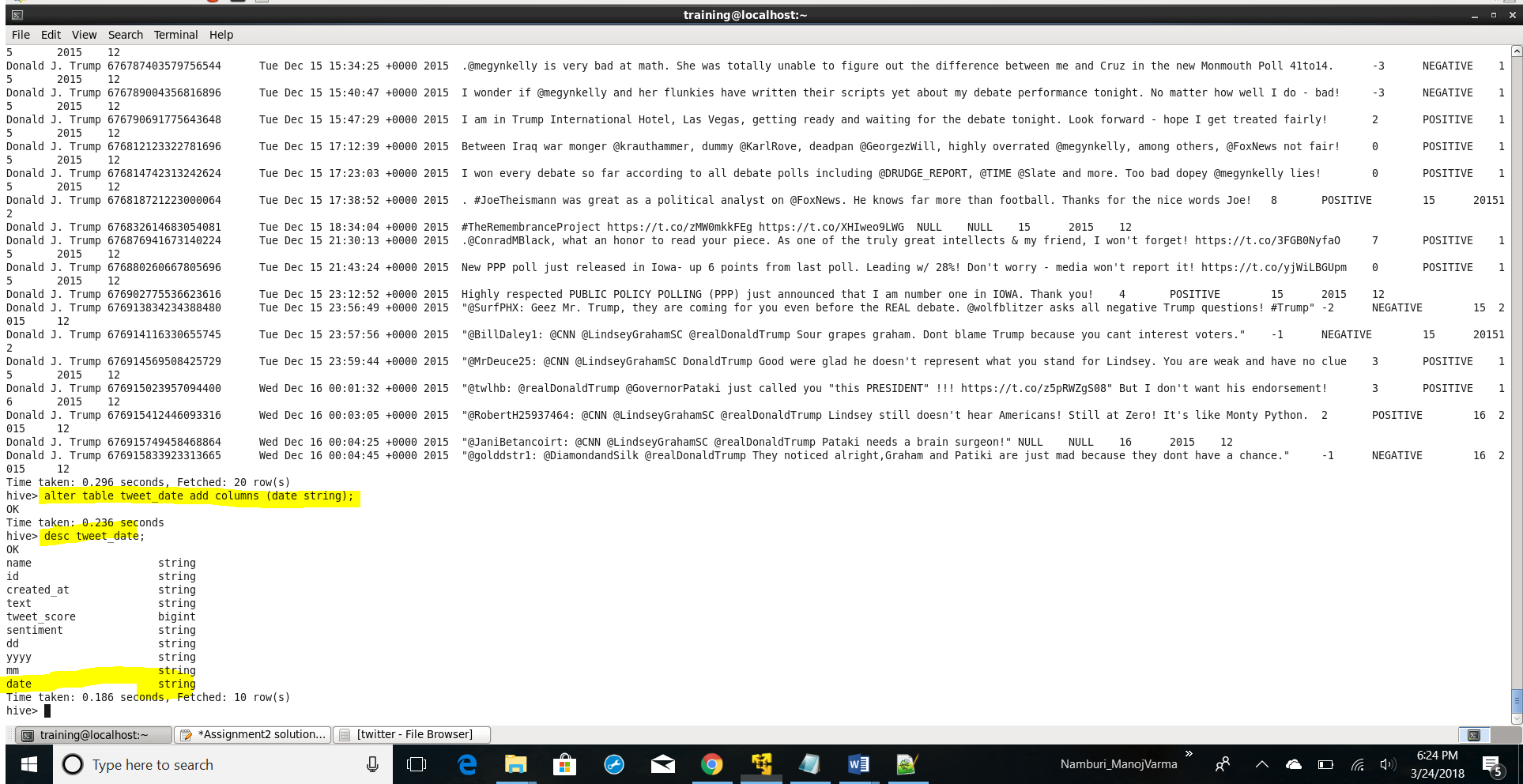
when substr(created\_at,5,3) = 'Dec' then '12'

end as mm

from tweet\_score;

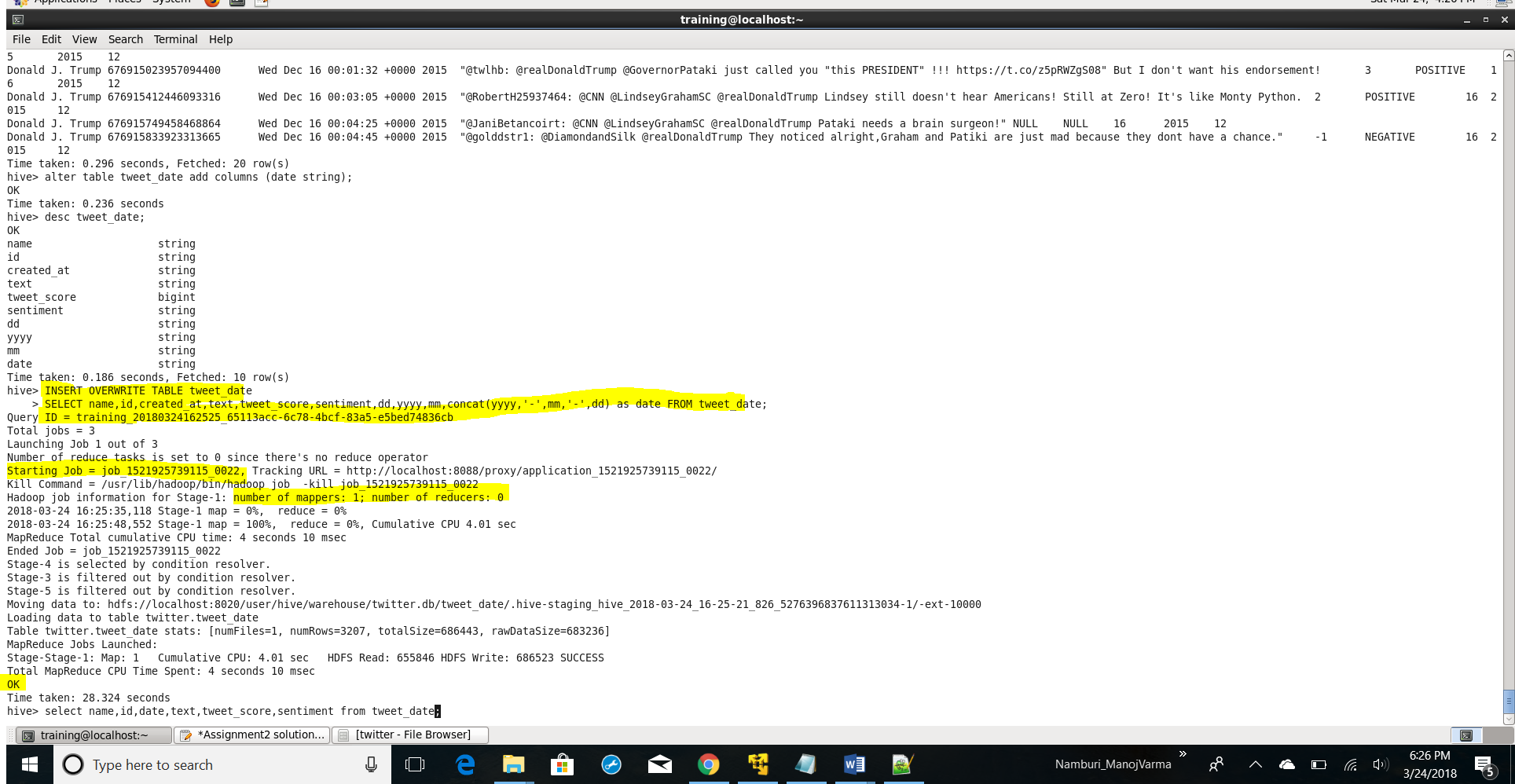


alter table tweet\_date add columns (date string);

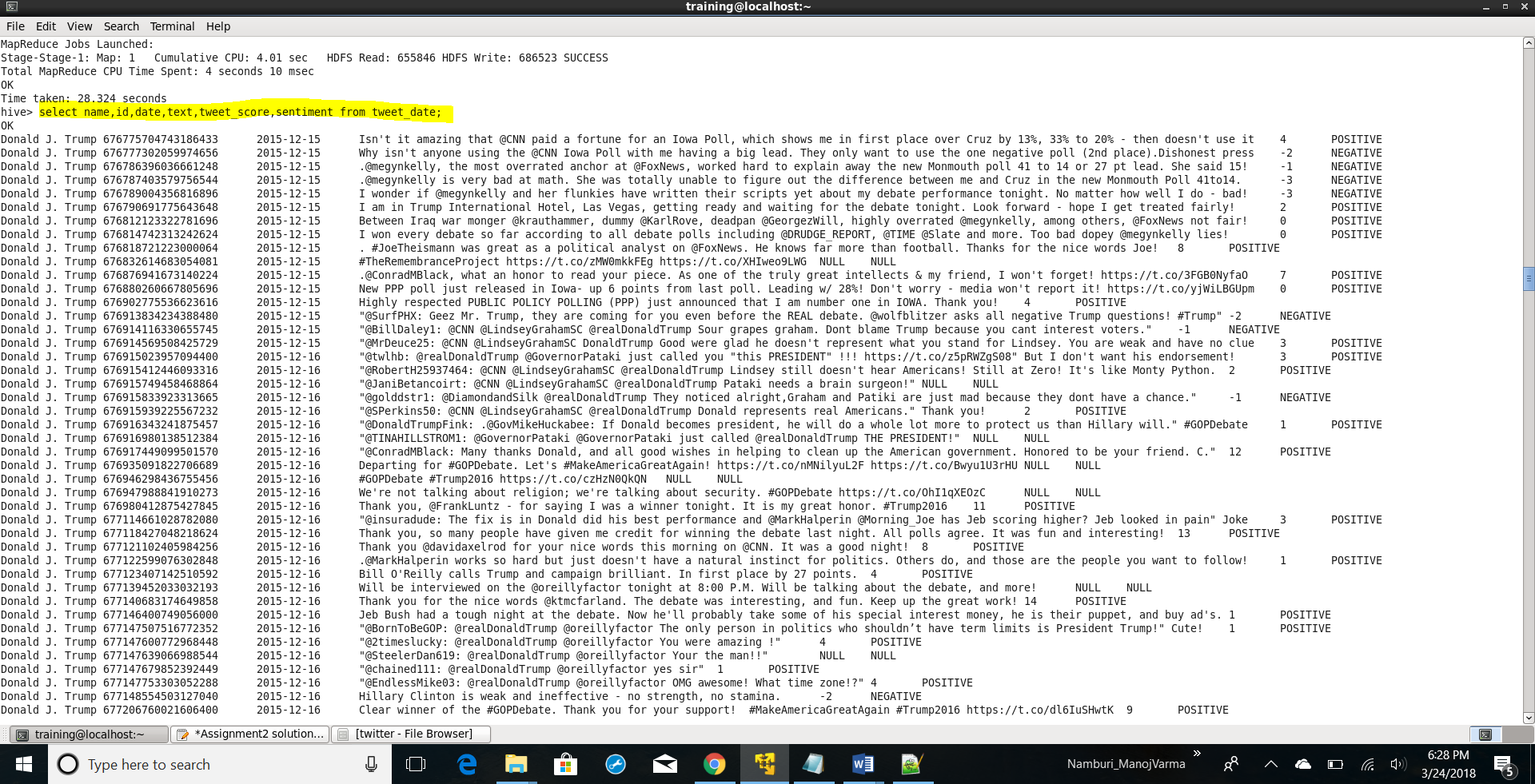


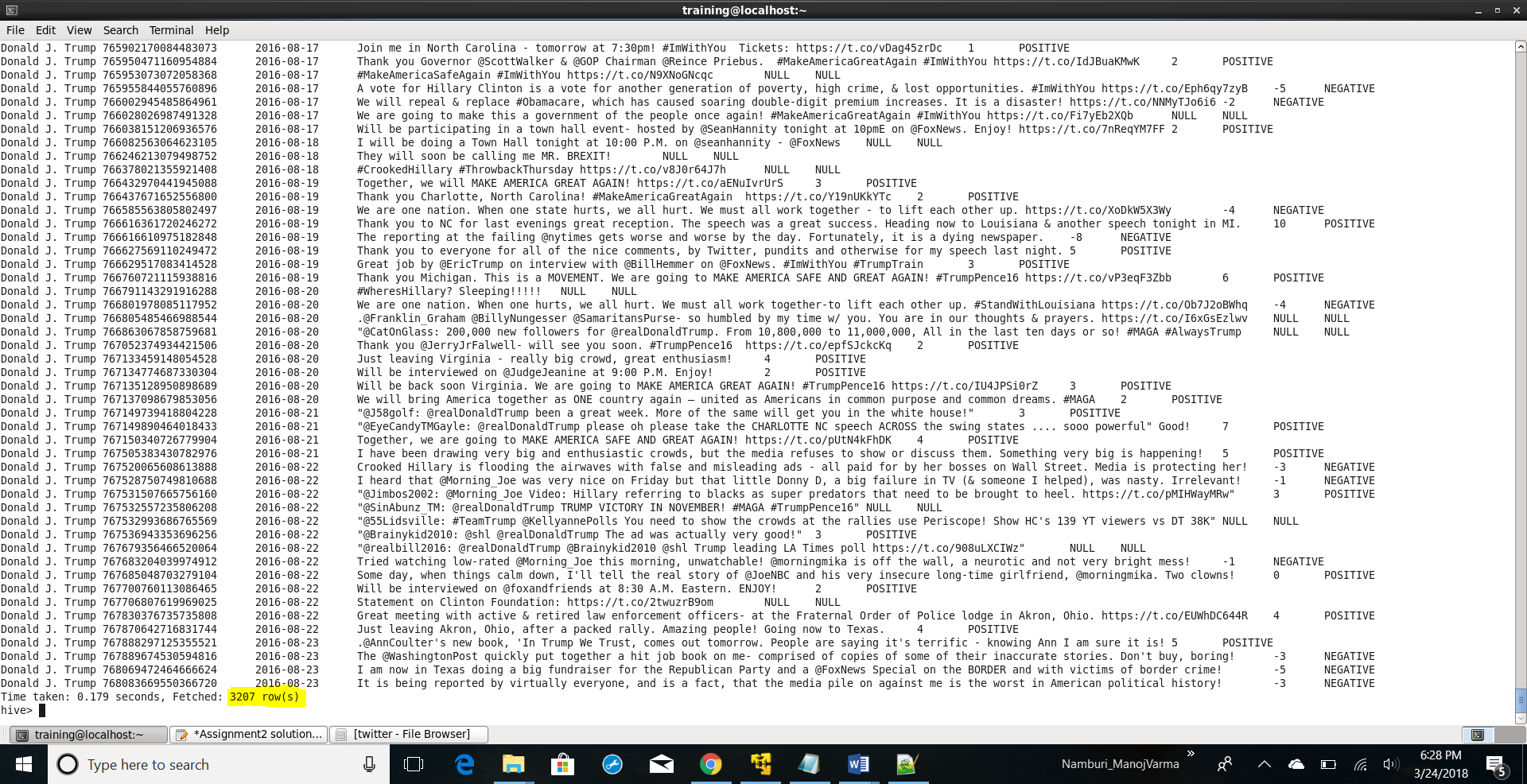
INSERT OVERWRITE TABLE tweet\_date

SELECT name,id,created\_at,text,tweet\_score,sentiment,dd,yyyy,mm,concat(yyyy,'-',mm,'-',dd) as date FROM tweet\_date;



select name,id,date,text,tweet\_score,sentiment from tweet\_date;





Question2

Do you find any problem in the way sentiment analysis was performed in the previous question? If so, how will you improve it?

Yes there is a problem in the way sentiment analysis was performed in the previous question.

Just by adding up the polarity scores of the words in the sentence we cannot decide the sentiment of the sentence.

Lets consider the following sentence: “My flight’s been delayed. Brilliant!”

From the dictionary if the polarity of the word Delayed is -1 and Brilliant is 3, the polarity of the above sentence becomes 2 which indicates positive sentiment

Most humans would be able to quickly interpret that the person was being sarcastic. We know that for most people having a delayed flight is not a good experience. By applying this contextual understanding to the sentence, we can easily identify the sentiment as negative.

The main problem when designing methods for the treatment of tweets is that they are highly informal texts, i.e. they contain slang, emoticons, repetitions of letters or punctuation signs, misspellings (done on purpose or due to writing them from mobile devices), entire words in capital letters, etc.

Handling of negation is a challenging task. Without the use of any negative word, negation can be expressed in many ways. English sentence to illustrate the concept of negation is given

I do not like the movie.

In this sentence, polarity of all the words that appear after the negation operator (such as not) is changed. But, this method does not work for the sentence given

I do not like the acting but I like the direction. …

In this sentence, scope of negation should be considered which is only up to but here. To solve this, polarity of all the words is changed that appear after a negation word until another negation word comes. Even then, there are problems. Consider the sentence below

Not only did I like the acting, but also the direction. …

In this sentence, due to the presence of “only”, polarity is not reversed after “not”. So, these type of combinations of “not” with the words like “only” has to be considered during designing the algorithm

The human language is complex. Teaching a machine to analyse the various grammatical nuances, cultural variations, slang and misspellings that occur in online mentions is a difficult process. Teaching a machine to understand how context can affect tone is even more difficult.

Without contextual understanding, a machine looking at the sentence above might see the word “brilliant” and categorise it as positive.

we employ a **rules-based process** to help better understand the ways context can affect sentiment.

**Rules Based process:**

Dictionary of operators, which can revert or intensify the value of the sentiment expressions. Operators can intensify the base value of expressions (very much, considerably), etc. or revert the value to the opposite (not, abolish) etc.

• Dictionary of stop-expressions — the list of multiword expressions containing sentiment words, but not expressing any overall sentiment, for example, foundation of effective politics etc.

The first group contains the following set of rules (referred to below as algo)

1.1. If an operator word is a part of a longer stop-word or sentiment expression, it does not act as an operator;

1.2. If a group of operators appears together, their scores are multiplied;

1.3. If there is unknown hyphenated word appeared in a text fragment, it is divided in two words and their scores are considered separately;

1.4. If there is a sentiment word sequence, and a negative word appears among them then the score of the whole sequence becomes negative, otherwise positive;

1.5. An operator is applied to the resulting score of a group of sentiment words.

The second group contains the following set of rules (referred below as rules). The rules were modified from:

2.1. If there is a question mark in a sentence, and the sentence does not begin with the words (why, for what), its sentiment score should be reduced;

2.2. If there is (if) in a clause, the sentiment scores of the words in this fragment that go after (If) should be reduced;

2.3. If there is whether particle in a clause, and there is no such words as almost/little/just/what/that/ unlikely/scarce/few/ just before whether, the sentiment score of the clause should be reduced;

2.4. If there is (would) particle in a clause then the sentiment score of the words in this clause, which go after (would), should be reduced.

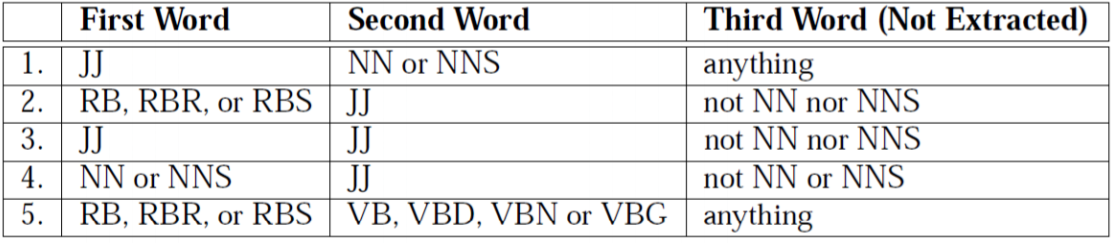
**The PMI-IR method**

● Using Pointwise Mutual Information (PMI) on data gathered using Information Retrieval (IR) techniques

● Yields real-numbered positive and negative scores for potentially any combination of words

● Requires WWW-sized unstructured training data resources

Extract descriptive 2-word phrases based on POS

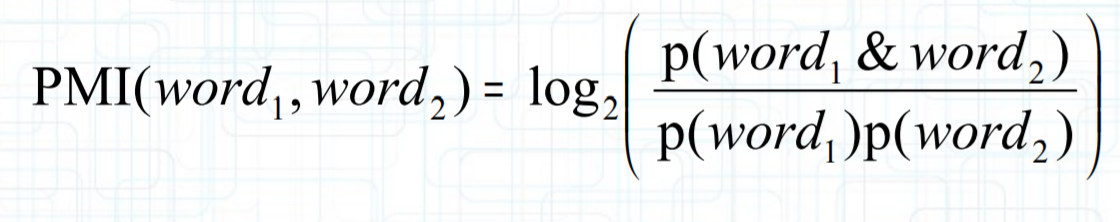


For each phrase, conducted Altavista searches using the NEAR operator, one with the word excellent and one with the word poor. ● NEAR operator (now discontinued) searched for the phrase occurring within ten words of the value word. ● Derive a score based on returned hit counts for each search and hit counts of the words and phrases on their own

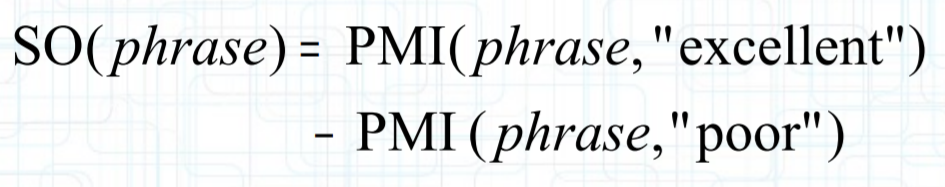
Calculating PMI

● word1 is the descriptive phrase, word2 is the value word

● p() is Altavista hit count (& is NEAR operator)



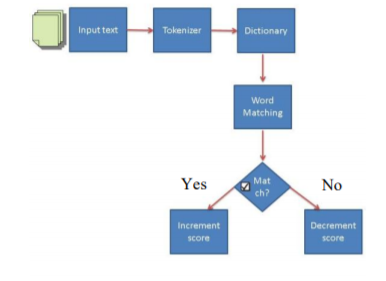
Deriving semantic orientation from PMI



Based on average SO of phrases in the review we calculate semantic orientation and find out the accuracy by using machine learning models

**Lexican based Approach:**

This technique is governed by the use of a dictionary consisting pre-tagged lexicons. The input text is converted to tokens by the Tokenizer. Every new token encountered is then matched for the lexicon in the dictionary. If there is a positive match, the score is added to the total pool of score for the input text. For instance if “dramatic” is a positive match in the dictionary then the total score of the text is incremented. Otherwise the score is decremented or the word is tagged as negative. Though this technique appears to be amateur in nature, its variants have proved to be worthy Fig. 1 shows the working of a lexical technique.



The classification of a text depends on the total score it achieves. Considerable amount of work has been devoted for measuring which best lexical information works. An accuracy of about 80% on single phrases can be achieved by the use of hand tagged lexicons comprised of only adjectives, which are crucial for deciding the subjectivity of an evaluative text. Other than the hand tagged lexicon approach, came up with a variant by utilizing internet search engine for marking the polarity of words included in work. They used two AltaVista search engine queries: target word + „good‟ and other target word + „bad‟. The score was evaluated by the search that yielded the max number of hits, which reported to improve the earlier accuracy from 62% to 65%. In Subsequent research, the scoring of words was accomplished by using the WordNet database. They compared the target word with two pivot words („good‟ and „bad‟) and found the **Minimum Path Distance** (MPD) between the words in the WordNet pyramid. The MPD is the converted to an incremental score, which is stored in the word dictionary. This variant was reported to yield accuracy of 64%. The author proposes another method which presents an alternative ,taking motivation from, by evaluating the semantic gap between the words simply subtracting the set of positive ones from the negative ones yielding 82% accuracy. Lexical analysis has a limitation: its performance (in terms of time complexity and accuracy) degrades drastically with the exponential growth of the size of dictionary (number of words).