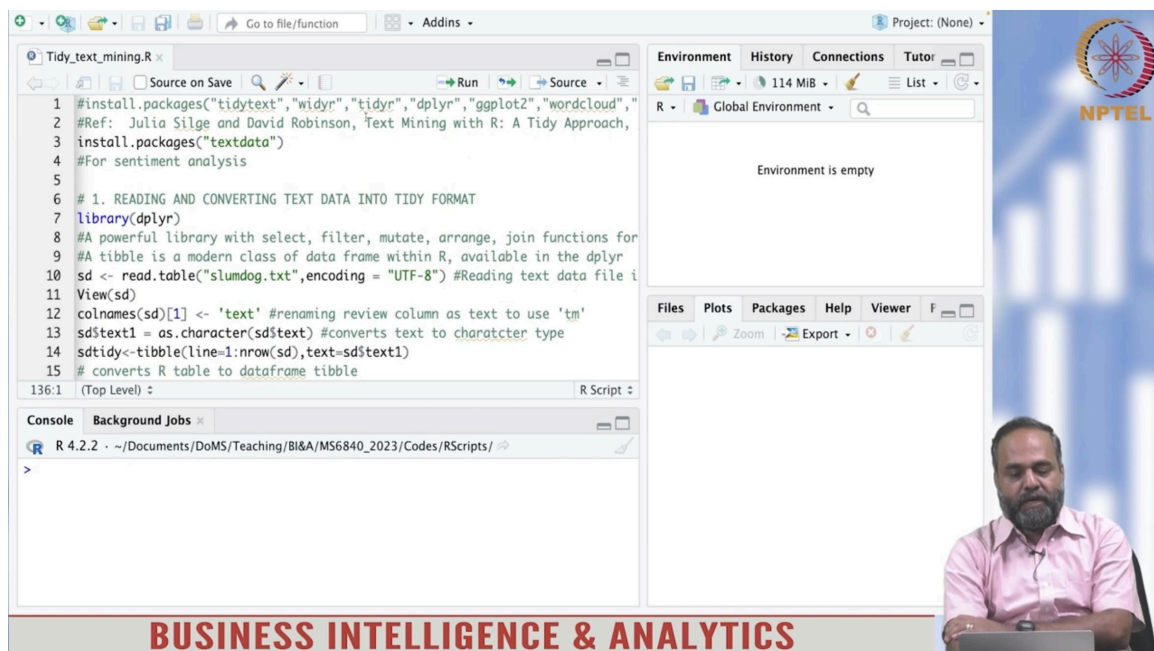


Course Name: Business Intelligence and Analytics
Professor Name: Prof. Saji.K.Mathew
Department Name: Department of Management Studies
Institute Name: Indian Institute of Technology Madras
Week: 12
Lecture: 46

Text mining using R- the case of a movie discussion forum

Hello and welcome back. So in this session, we are going to work on a short problem. So, I would say an interesting problem because I believe you all connect with the world of movies, entertainment. So, therefore the data is taken from a portal related to that. And my reference for this session, the reference book is Julia Silge and David Robinson's book on Text Mining with R, a tidy approach because it is a tidy tibble as a data structure instead of table is what I am going to use in this particular exercise.



The image shows a screenshot of the R Studio interface. The main window displays an R script with the following code:

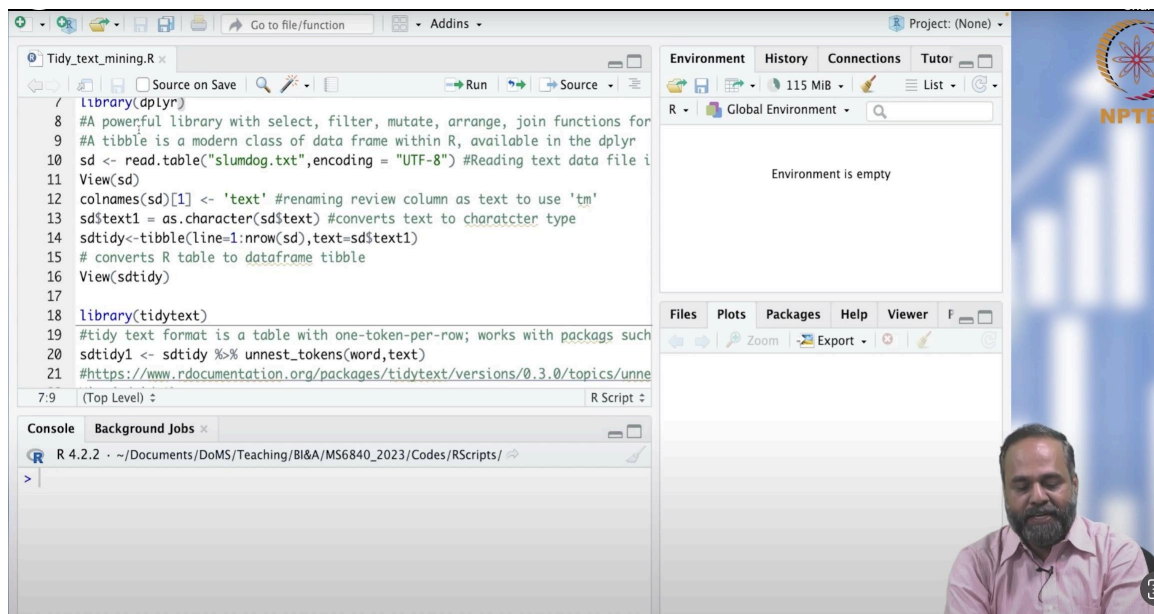
```
1 #install.packages("tidytext","widyr","tidyr","dplyr","ggplot2","wordcloud",
2 #Ref: Julia Silge and David Robinson, Text Mining with R: A Tidy Approach,
3 install.packages("textdata")
4 #For sentiment analysis
5
6 # 1. READING AND CONVERTING TEXT DATA INTO TIDY FORMAT
7 library(dplyr)
8 #A powerful library with select, filter, mutate, arrange, join functions for
9 #A tibble is a modern class of data frame within R, available in the dplyr
10 sd <- read.table("slumdog.txt",encoding = "UTF-8") #Reading text data file i
11 View(sd)
12 colnames(sd)[1] <- 'text' #renaming review column as text to use 'tm'
13 sd$text1 = as.character(sd$text) #converts text to charatcter type
14 sdtidy<-tibble(line=1:nrow(sd),text=sd$text1)
15 # converts R table to dataframe tibble
```

The console shows the R version: R 4.2.2. The environment pane is empty. A video inset in the bottom right corner shows the professor, Prof. Saji.K.Mathew, sitting at a desk. The NPTEL logo is visible in the top right corner of the R Studio window.

BUSINESS INTELLIGENCE & ANALYTICS

So, this has got lot of traction today, the tidy approach to text data mining. And this particular book is something you can buy or entire text or content is given online and I have given a reference to that here, in the code itself. So, I will be using the R studio which is a platform which runs the R and I have updated R and installed these packages. The procedure is given right here if you want to. If you have not installed you should first install these packages and subsequently invoke the libraries or call these libraries when you run the functions from these libraries.

So, the data actually pertains to a discussion forum. As I said when the movie Slumdog Millionaire was released, there was intense discussion on the movie and of course, the movie is lot of entertainment and people liked the movie. But there was also a complaint that India was depicted as a country of slums and that is a major theme of the movie and directed by British director, but it has a lot of Indian artist in it and so, a lot of confusion and discussion. So that is why I picked this particular discussion forum data. And also you know this is something that people from different countries can understand because this is a academy award winning movie. Alright, so my effort is to look at this data and also get some insights about what are people talking about, are there good words or bad words or ugly words and so on and how they correlate etc.



```
Tidy_text_mining.R
7 library(dplyr)
8 #A powerful library with select, filter, mutate, arrange, join functions for
9 #A tibble is a modern class of data frame within R, available in the dplyr
10 sd <- read.table("slumdog.txt",encoding = "UTF-8") #Reading text data file i
11 View(sd)
12 colnames(sd)[1] <- 'text' #renaming review column as text to use 'tm'
13 sd$text1 = as.character(sd$text) #converts text to character type
14 sdtidy<-tibble(line=1:nrow(sd),text=sd$text1)
15 # converts R table to dataframe tibble
16 View(sdtidy)
17
18 library(tidytext)
19 #tidy text format is a table with one-token-per-row; works with packages such
20 sdtidy1 <- sdtidy %>% unnest_tokens(word,text)
21 #https://www.rdocumentation.org/packages/tidytext/versions/0.3.0/topics/unnest
```

Environment History Connections Tutor
R - Global Environment - 115 MIB
Environment is empty

Files Plots Packages Help Viewer
Zoom Export

7:9 (Top Level) R Script

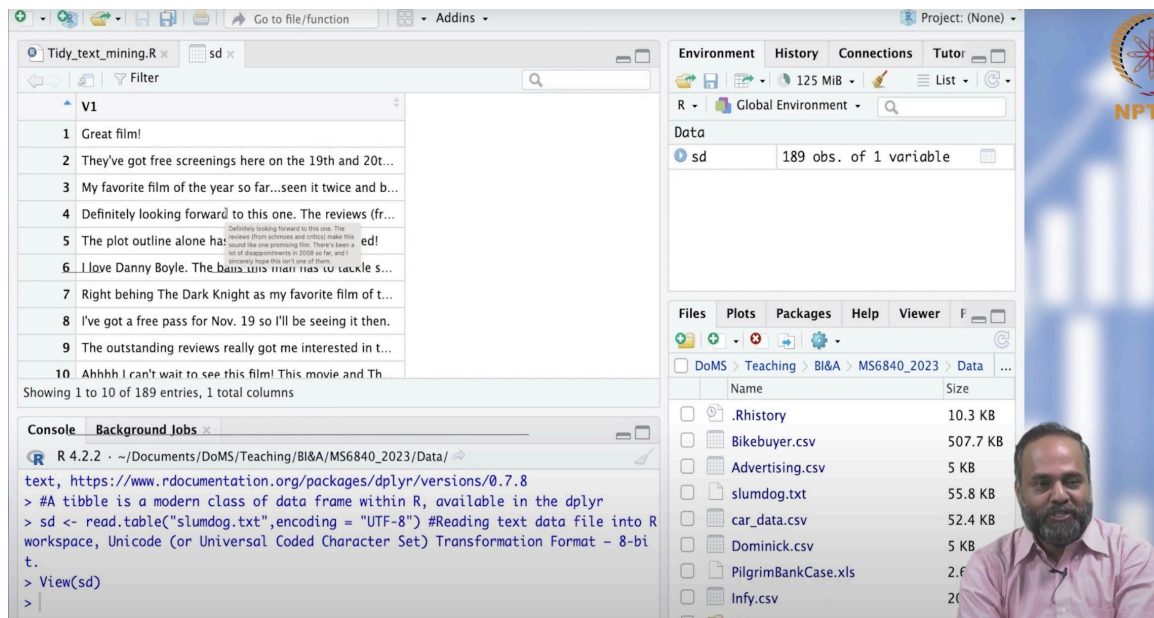
Console Background Jobs
R 4.2.2 - ~/Documents/DoMS/Teaching/BI&A/MS6840_2023/Codes/RScripts/

So some basic level of text analysis we can do, using this data. So, first step, line 7 I am going to call the library dplyr, d p l y r which is a library useful for working with tidy text and you can read this data and then manipulate this data, create a tibble which is a data structure I said for tidy text analysis, all that using dplyr. So, I am running code line by line, I can run the code as a whole, I can run line by line. So, I am just running this codes line by line or before that, you know this is R. So, I need to define the path.

So the first thing is my data is stored in a particular folder in my computer. So, I am going to the file section and I am going to point or define my path here to the folder where my data is stored. So, for me that data is in MS6842 in the data folder. So, my workspace files pane is pointing to a specific folder where the data is contained. I notice that this particular procedure is similar to that of matlab.

So, there is my file pane now points to that particular data folder and now I say, I go to my session and say what is my directory. I say my directory is the files pane location. So, therefore, data will be read directly from the files pane which point to the data folder where my data is stored. So unless you do this data, you will have to write the complete path if you do not do this. Instead of that you can just point to the folder. So, let me do this and then start my exercise of the library is invoked and subsequently I am reading slumdog.txt which is there in my file pane. You can see that slumdog dot text is right here because my file pane has, points to that particular folder and that is my path. So, the data is read as a table into the SD data frame SD, SD is the name of the variable or data frame.

Now, you want to see how this discussion forum data looks like. It is data is read, something some issue here, I do not know I did not run the previous command, that is why. See you can see my data is this, which is available in text format, I read into it and you can see, these are comments or posts by different people each post is a document. Each post is a document, somebody says great film. I should not be reading all these comments because there will be lot of unparliamentary comments here. So, it is an uncensored data and do not complain about the data because it is not my data. It is a discussion forum data.



Data taken uncensored and so how many documents are there? If you go down, it is 189 documents, 189 post, 189 post taken as it is, that is my data. Read in viewed, so now I am going to give a title to the column. So that is a simple labeling and then I am defining the data type as character type to take care of some of the issues with the libraries and the functions. And now I am defining a tibble, tibble is a particular data type or a data

structure in dplyr, in tidy text, it is tibble, it works in tidy text well. So, the data format I am converting as tibble or my text one, which I define now which contains the text data in document format is converted to a tibble or a table you can call it a table in tidy text format a tibble is nothing but a table in tidy text. Maybe I did not execute a line that is fine. That is done and now let me see what the tibble does. Well, you see a tibble now has two columns. It is a two column table. One is a line. It is titled as line, other column is titled as text. That was a original table, original column.

So idea here is in a tibble, each row has an identifier. That line is nothing but an identifier for a document. So document one, document two, document three, it gets actually identified. It is like a key added, a tibble adds a key to a table, to identify each row. So each document or each document here, so each document has a document identifier. For example, great film doc one, they have got free screenings and so on, document two. So, that is a key added. Let us have that understanding and more. Now I am calling library tidy text, which is another library for tidy text, invoked and now you see the command line 20 s tidy one, which is another tibble that I am creating. This s tidy, there is a pipe operator, you know this is a more recent development in programming. you s tidy is the source file pipe unnest tokens word text.

The screenshot shows the RStudio interface. The main window displays a data table with two columns: 'line' and 'word'. The first 10 rows are visible, showing document identifiers and their corresponding words. The console shows the R code used to create the tibble 'sdtidy1' from a source file 'sd' using the 'tidytext' package's 'unnest_tokens' function. The environment pane on the right shows the current data objects: 'sd' (189 obs. of 2 variables), 'sdtidy' (189 obs. of 2 variables), and 'sdtidy1' (10299 obs. of 2 variables). The file explorer shows the project structure, including a 'Data' folder with various files like 'Bikebuyer.csv', 'Advertising.csv', and 'slumdog.txt'.

line	word
1	1 great
2	1 film
3	2 they've
4	2 got
5	2 free
6	2 screenings
7	2 here
8	2 on
9	2 the
10	2 19th

```

R 4.2.2 - ~/Documents/DoMS/Teaching/BI&A/MS6840_2023/Data/
> #tidy text format is a table with one-token-per-row; works with packages such as d
plyr, tidyr, ggplot2 etc
> sdtidy1 <- sdtidy %>% unnest_tokens(word, text)
> #https://www.rdocumentation.org/packages/tidytext/versions/0.3.0/topics/unnest_to
kens
> View(sdtidy1)

```

So, from the text column, the tidy text document will be unnested into tokens. We discussed tokenization. A document consists of tokens and tokens are what? Tokens can be words, tokens can be punctuations, tokens could be numbers and so on. But here when I use the command unnest tokens which is a tidy text command, word comma text means, word will be the unit word will be the token. Unnesting will be done based on words and comma. So, it will be applied on the data text, the column text. So, s tidy has

a column called text. So, this is the unnesting operation and sorry, I did something wrong.

Now, I want to view how a tidy one looks like. Now you see, after tokenization the tibble looks like this. Line means it is the document number. So document one, document one has how many tokens? Great has a token which is a word and film has a token which is word. The punctuation has gone because it is, the basis is word unnest underscore tokens word, not punctuation. So, it removes the punctuations. So, document one great film, document two has so many words, but all of them has the identifier two showing that these, each of this word belong to the document two.

So, it has kept a identifier. So, that is the point. So, it is structuring the unstructured data in one sense, we can see it is adding an identifier. So, the word got may be there in other documents also, but here the word got belongs to the document two.

Let us now move now there is a command, interesting command count word. So, in tidy text there is a count command. Using that you can count the number of words and here itself we have used some descriptive analysis. You see the corpus has the word the, which has the highest occurrence, 588 times, the word the is occurring in the corpus and and is occurring 294 times, i is occurring 273, of occurring 270, but so what? Who is interested to see how many times d has occurred, of has occurred and so on and so forth? You feel disappointed and that is why we talked about stop words. We do not want these words to be there in the analysis. This words should be removed from the count. So, then we come to the section of stop words.

I created this document, this script using the pandemic section. So, I have structured it in a way, I had timed those days. So, this is created in a way you can, section wise the code scripts are arranged. So, you can easily follow this. So, you can see that in stop words what you see is, stop words is a lexicon in the in the tidy text library and we are loading that first.

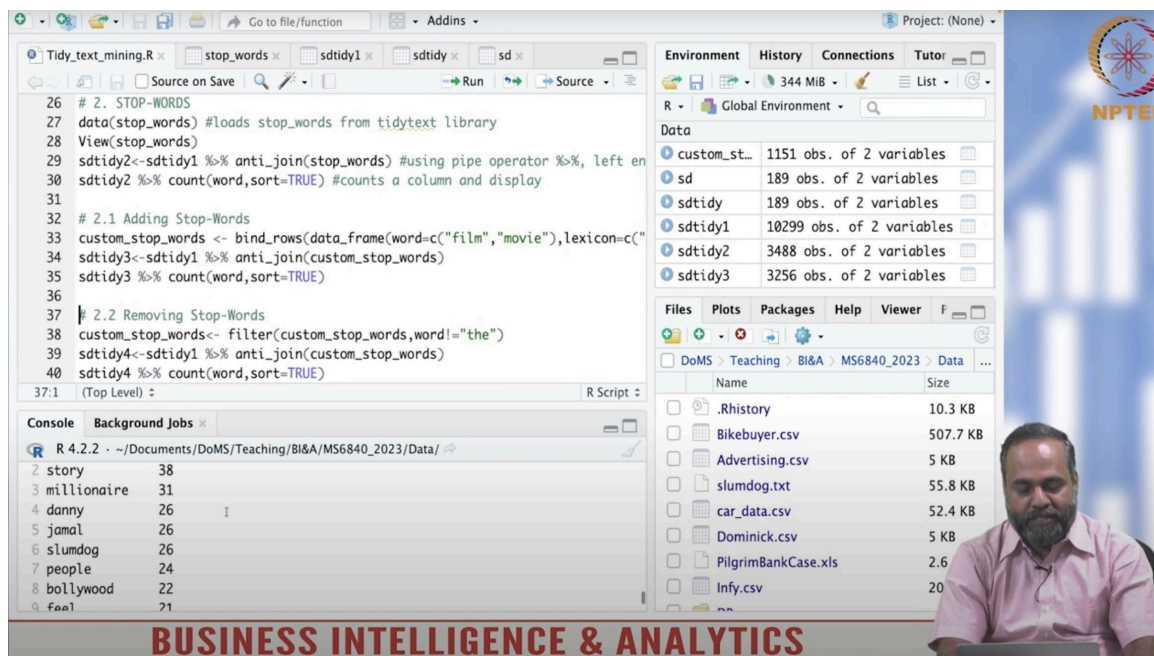
And if you want to see what are stop words like, we can view that. The name of the lexicon, stop word lexicon is smart and you can see that there are 1149 stop words. And what are the contents of the stop words? A, as, able, about, above, according, accordingly, across, actually, afterwards, again, against. It is a collection of commonly occurring words which we are not interested to sort of analyze for descriptive analysis. So, that is the stop words.

Now, you see you want to remove the stop words from your original corpus. How is that done? You anti-join, there is a command called anti-join. Anti-join means those words will be, this words will be removed from your original corpus. You see the anti-join

command using the pipe operator. So now, let us look at the counts.

The meaningless, the useless words are gone as we say. So, now let us look at which are the popular words in the discussion forum, interesting. The word film occurs 163 times. Of course, the discussion is about films and naturally this is going to come.

The word movie is 69 times. Again you are disappointed, like I do not want to see this. If I have to create a word cloud I do not want the film or movie showing so much importance because that is understood that this is there. You may want to include the words movie and film as stop words because you do not want them. So, then subsequent comments actually, subsequent lines of code does that. If you want to add an extra word to stop word, you can do it like that. This is about adding them and then anti-joining with the added words.



The screenshot shows the R Studio environment. The script editor contains R code for text mining, including loading stop words, creating data frames, and using the pipe operator. The console shows the output of the `count` function, listing words and their frequencies.

```
26 # 2. STOP-WORDS
27 data(stop_words) #loads stop_words from tidytext library
28 View(stop_words)
29 sdtidy2<-sdtidy1 %>% anti_join(stop_words) #using pipe operator %>%, left en
30 sdtidy2 %>% count(word,sort=TRUE) #counts a column and display
31
32 # 2.1 Adding Stop-Words
33 custom_stop_words <- bind_rows(data_frame(word=c("film","movie")),lexicon=c("
34 sdtidy3<-sdtidy1 %>% anti_join(custom_stop_words)
35 sdtidy3 %>% count(word,sort=TRUE)
36
37 # 2.2 Removing Stop-Words
38 custom_stop_words<- filter(custom_stop_words,word!="the")
39 sdtidy4<-sdtidy1 %>% anti_join(custom_stop_words)
40 sdtidy4 %>% count(word,sort=TRUE)
37:1 (Top Level) :
```

The console output shows the following word counts:

Word	Count
story	38
millionaire	31
danny	26
jamal	26
slumdog	26
people	24
bollywood	22
feal	21

So, I am creating s tidy 3 with the added stop words and counting again and now I am again looking at the counts. Well, now it makes a little more sense. Boyle, Danny Boyle Boyle is a director. So he is having count of 48, story 38, millionaire 31, Danny 26, Jamal 26, slumdog 26, people 74 and so on. Now, I am getting to see what is the discussion about. Well these are the most frequently occurring words, just the frequencies. And suppose this is incidental. Suppose you want to remove some stop word. You know suppose you want the word I is important. You want to retain that word or you want to retain the, instead of the let me make it I and I want to include that, you can do that.

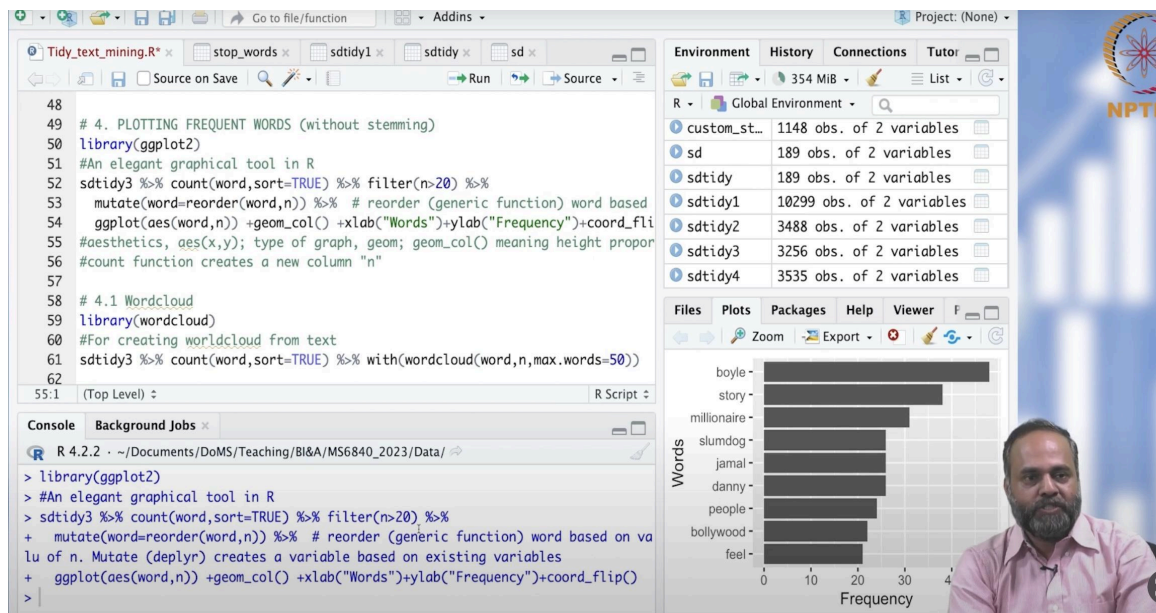
So, I thought this will be useful to you when you do actually text analysis. I have included I, it is occurring 279 times just to sort of learn the method for stop words inclusion and exclusion.

Now we discussed stemming. We discussed stemming. Now, we are implementing stemming here. There is library Snowball C. Snowball C in R is for stemming. So, commands for stemming are available in Snowball C. So, we are actually getting that library, Snowball C here. And so, you can see the, there is a command mutate where you can add a line or sorry, add a column to a tibble in steady text using the mutate command.

So, I am stemming the words that is from s tidy 3 and then counting them. That is line 33. Now stemming is as I said, to include all forms of a word be it present, past, past continuous, present continuous whatever. Now you see the count of the word love. Let us go before. How many times the word love occurred before stemming? 20 times.

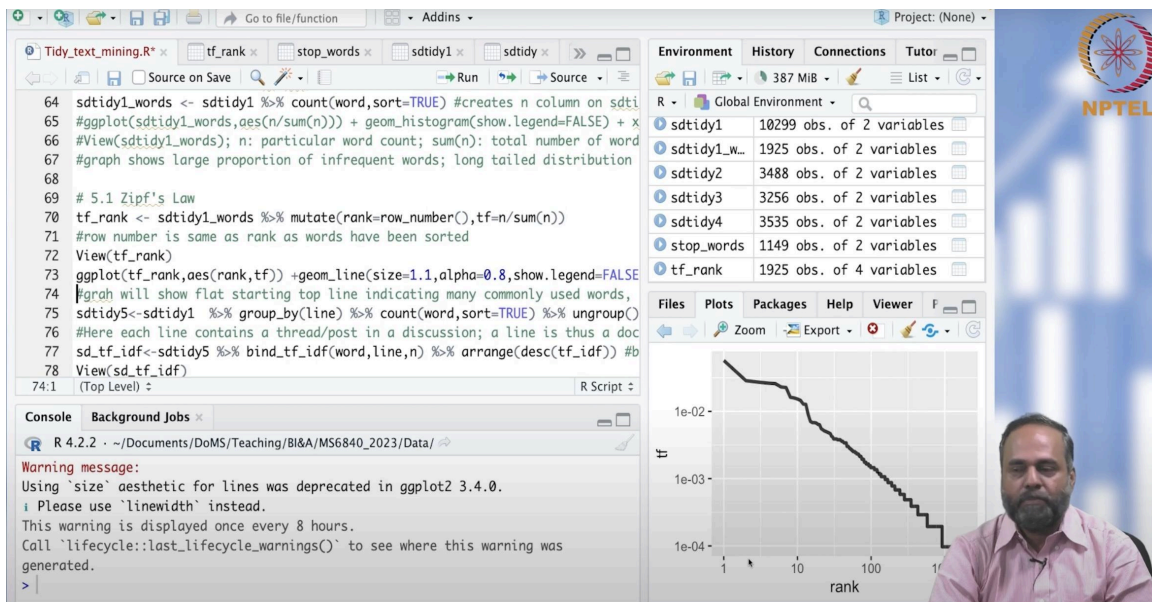
But after stemming, how many times the word love is occurring? 35 times. Because it has captured all forms that, of that word love. You do not miss that. So, you see the usefulness of stemming particularly when you are looking for certain sentiments or certain words which are important to you. So, that is the use of stemming.

Now let us plot these words in the order of their importance. There are, I again refer to Silgay's book and in that sequence, I am giving you the instructions. So, ggplot is like matplotlib in python, is a very useful library for plotting in R. So, I am using ggplot2 here. So, the library and the words are actually plotted here in the, as bar chart.



So, you can see the most important words after removing the stop words, importance in terms of word counts, these are the words. So what is the story if somebody ask you in this particular corpus, what is going on? So, have a plot of this kind. Well what this is what is going on. This is what people are talking about. It is a first impression. And all of you today, use word cloud. There are very creative forms of word cloud available. So, R has a word cloud library and you can use that for creating a word cloud where the important words have higher font size and unimportant words have lower font size. So, that is the way it organizes. It gives you a sense of important words in a cloud format, instead of bar chart format.

These are clouds you can further improvise on. So, I leave this section. Zipf's law is explained in Silgay's textbook and essentially if you look at a corpus as a whole, there will be a lot of unimportant words, any text, any book or any collection of documents will have a lot of unimportant words. But like the words, like the a, accordingly, able, like, this, that etc. You know those words are not very important. But that will be the most or the largest number, in terms of words that are used. So Zipf's, this particular graph actually helps you understand or compare two corpuses in terms of occurrence of common words.



I will, let me first plot this and show you. Here it calculates the term frequency. We discussed what is term frequency. The term frequency, we have not used stop words here. So, we are doing a s tidy 1. So, the term frequency is actually the tf, the number of times that word occurs in a particular document divided by the number of words in the document. You can see that here. So, the word the has the highest tf and so on. So, that is the tf that is calculated here. And there is a plot using tf here. This is the Zipf's law,

which is explained in your text book, that is the rank of words. The commonly occurring words will have high rank in terms of tf because they are the most occurring. The word the may be the highest followed by and etc.

So, they all come here in terms of rank. And then you can see that this particular part of the graph shows, you know the popularity of the common words. The longer this line or the vertical this line, if this line is occurring for a long region, it shows that a lot of common words are very popular in a given corpus or a document, depending on the unit of your analysis. Ideally it says this length should be as small as possible, so that you know, you have a lot of content that is useful in a document.

That is all. Let us move on. Now we are going to look at the tf idf together. Importance of words. We saw that more than tf, what is more useful to understand the importance of a word with respect to documents is the tf idf. And you can see that tf idf is with respect to documents. This is document 117. So, this is the word pa. It is occurring in the discussion. That is having the highest or pa is having the highest tf idf measure, for the importance of words. The word rubbish is having the next highest tf idf which is 2.62. Both are actually the same. And this pertains to the document 117. And this has got, among all the documents this is the highest tf idf. Among all the documents this is the highest tdf idf that is occurring in 117th document and the value is 2.62. So, if you have to have a relative sense of which is the most unique and important word and where is it occurring, you go there, to the document 117.

And these three words have been spotted as having the highest tf idf. Rubbish, exceptionally, strange, perfectly. These are unique, at the same time very important words. Hard to interpret why it is so in this discussion. So, I leave it there.

Now bigram analysis. We already have seen that words occur together in documents that is, you know co-occurrence. Words also occur in a sequence that is the n-gram. And these occurrences are important and the measures that tell us how often they occur together are also insightful in text mining. So, we again start our tokenization using n-grams. So, n-grams is the basis for tokenization here, that words have high count in terms that they occur in a sequence together.

And when I say n equals 2 or when I give the argument n equals 2 here, for n-gram tokenization, it is bigram. Then it un-nests the documents or tokens documents as bigrams. And let us see how the bigrams look like.

How many bigrams here? 10,000 or so. And this is line 1, great film, they've got, free screenings, free, here on, on the, then all kinds of like, here on, they've got, you know

they've is, you know when you have an apostrophe it is taken as one word. So, you see a lot of bigrams which are again like, not useful for analysis. They should be filtered out. So, it is a simple bigram analysis, but you can apply stop words to bigrams as well.

That is what is done next, you can see. Let us move on. Let us count co-occurrences after removing stop words. So, there is a procedure for removing stop words from bigrams and subsequently you see, you look at the word counts. So those, after stop words are removed you can see there is more meaningful bigrams like Danny Boyle, Slumdog, Dev Patel, Love Story. Numbers are not removed. So, they are also shown there and how many times they are occurring. So, the bigrams which are meaningful bigrams.

Now, you can join them also, that is just for demonstration. Of course, I am calculating the phi coefficient here and WIDYR that is a library which has this pairwise correlation phi coefficient and all that in the tidy text format.

You can call that and calculate the pairwise correlation. No problem. So, that is what we are doing here. Let us move on and I am calling that and then look at pairs. So here, please keep in mind when you go for the phi coefficient, it is not bigram or n gram, but words that co-occur in a document, that is different from occurring in a sequence. Here in phi coefficient, it does not have to be in a sequence. It is only that two words have occurred together in a document and how many times they have occurring. That is what it is finding out, document wise. Alright.

So you can see Boyle, Danny, movie, film they have occurrence together 22 times so, and so on. And you see 105th line. What I am trying to do there is, which are the words to which love is associated with. The two words. So when word love occurs, which are the other words that occur? So, people love films, people love story, people love Danny, how many times? Love movie, love Jamal. You see the usefulness of it. You fix one word and see what are the other words and their frequencies associated with it. So it is a simple but very useful analysis.

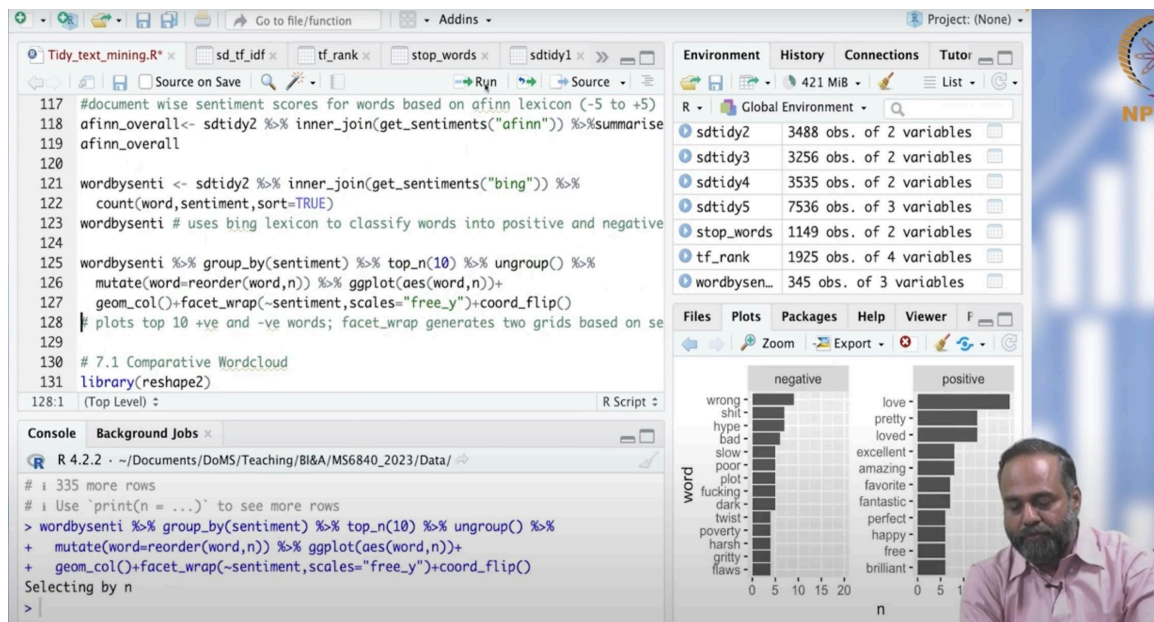
So, you can actually take a word gali. Kis kisko gali diya hai, dekhna hai toh gali lelo. And then see which are the people associated with that. So, or if there is a bad word or a good word or all that, you can actually use pair wise association. In a forum like this, there are lot of abusive words. I just wanted to caution you that. So phi correlation, phi coefficient is calculated there. Instead of the counts, you can actually look at the phi coefficient.

So direction Boyle's, that has the highest co-occurrence with a phi coefficient of 0.651 followed by Danny Boyle, Boyle, Danny same, that is very trivial. Slumdog millionaire,

also understood. Saleem Jamal. But some of these combinations when you go through it, may be insightful which are the words that are co-occurring.

So I also, I am seeing Danny and its phi coefficients with others Danny and Boyle, Danny and Life, Danny and Time, Danny Jamal and so on. But value of positive, but low. So, you can see that the co-occurrence could be analyzed using frequencies, co-occurrence should be explored using phi coefficient. So you can, you get to see, you take one word and see what are the other words which are strongly correlated. So, that is a useful analysis with text.

And then there is sentiment analysis. This is a simple procedure and I, in today's world most of you do this sentiment analysis, either using AFINN or NRC or any of this lexicons which are available. And I just run this code and to give a final plot of the sentiment analysis. So when you use, say for example AFINN, you get to see the words and their values. So, you get a sense of the scale of these words and which are the ones which are used and so on.



Let me get into plot. So, you get an overall sentiment also. Leave that, you can do it on your own and how many negative words, positive words and so on. These are different lexicons. You can plot it, positive words and negative words separately and which are the top ones that gives you a sense. Love words, love is there as a top word here. And that is having the longest and that is having more importance than the negative words.

So, positive words apparently is higher than the negative words in this particular discussion forum. And well, you can look at this. I have not censored it as I said, but you

can see all words here. And you see both love and hate for the movies plotted together as word cloud and that, you get a sense and you can see that love dominates and so on. If you are doing analysis or if you want to visualize overall sentiment in a more visual form, you can use these scripts which are given.

Well, I have not, I have mostly followed Silgay's textbook of course, customized it for the given context and dataset. And I will strongly recommend that book along with you when you do, if you are using tidy text to do a very fundamental level text analysis for frequencies, correlation and visualization. That is all we have done here. This is bag of word analysis, not semantic analysis. That is beyond the scope of our syllabus.