

Marketing Analytics
Professor Swagato Chatterjee
Vinod Gupta School of Management,
Indian Institute of Technology, Kharagpur
Lecture 57
Text Mining and Sentiment Analytics (Contd.)

Hello everybody. Welcome to Marketing Analytics course, this is Doctor Swagato Chatterjee from VGSOM, IIT Kharagpur who is taking this course for you. We are in week 11 and this is session 2 and we are discussing about Text Mining and Sentiment Analysis. So till now I have discussed about how to do sentiment analysis using Naive Bayes algorithm which is a very easy type of algorithm, but there are other better algorithms which are also possible.

Like you can use random forest or you can use SVM, you can use decision tree with the same document or matrix you can predict the y variable. Now many people, many other researchers have actually done that and in different context like I have taken only for hotel review they have done in different context and they have collected this data created corpus and then labeled it as positive or negative manually sometimes.

And then found out that which words are associated with positive review and which words are associated with negative review. They have done negative not only review negative sentiments they have done it for quite sometimes. Now after doing it for quite sometimes there are various kinds of researchers who have created various libraries. Library means these are actual text library which is like your Oxford library and kind of stuff.

Why there are lots of words are there and instead of meaning of those words which you can normally find out in English libraries. They have the sentiments course of those words. So if this word is occurring how much you can consider in the sentiment of this particular text. So some library has analyzed it and given in a positive and negative this much classification has been done.

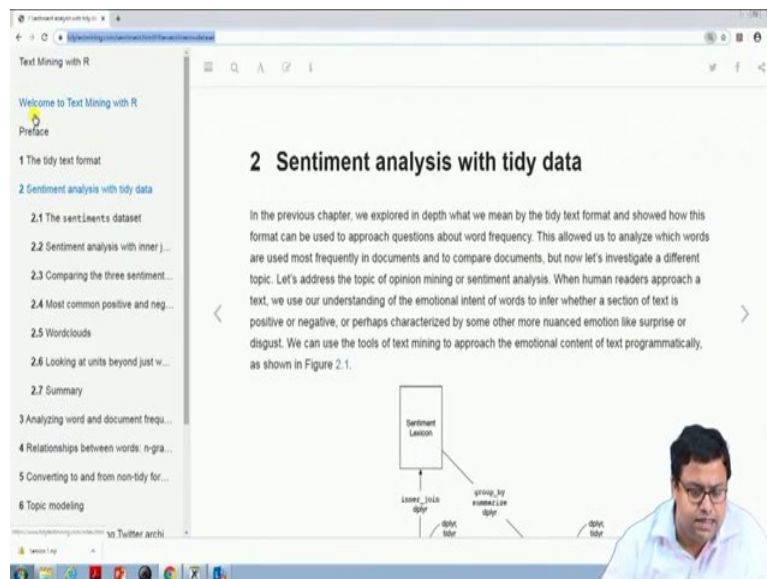
Some guy has given positive also how much positive means in your - 5 to + 5 scale they have created a continuum and they have given a rating for each of the words. There are other libraries where not only sentiment has been told, but also the emotions that is expressed as also been told like anger or frustration or disgust or let us say happiness or joy. So these kinds of emotions also were told.

So these libraries are already there and often times they were good. Ideally you should not use all these libraries in wrong context, but for a straight quick decision-making kind of an approach we can use this kind of readymade libraries.

You have to understand that we are marketing guys, we are not computer science guys. So we will not be trying to create new libraries that might not be our goal. Our goal will be using the algorithms and techniques that has been offered by the statisticians and computer science people in the context of marketing.

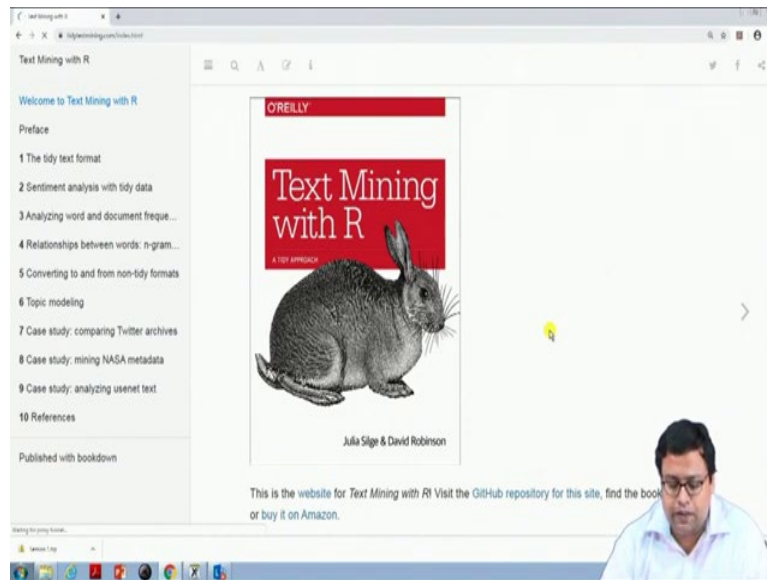
So rather than bringing in and creating your own library a marketing analytics professional should majorly focus on how I can use somebody else dictionary and create my own insights and how I can use those insights in marketing decision-making. So this is something that I will be doing in this particular session and I have no code for this particular session, but we have is a website. So if you see this particular website the link has been shared.

(Refer Slide Time: 03:35)



So if you see it is a tidy text mining there is a library for that also tidy text and this is something that we will be going to discuss in this particular class. So what I will do to discuss in this particular class is I will ask you to go this to this. This particular link has been shared in a text file in the files that you got and this is something that we will be discussing this is a book basically.

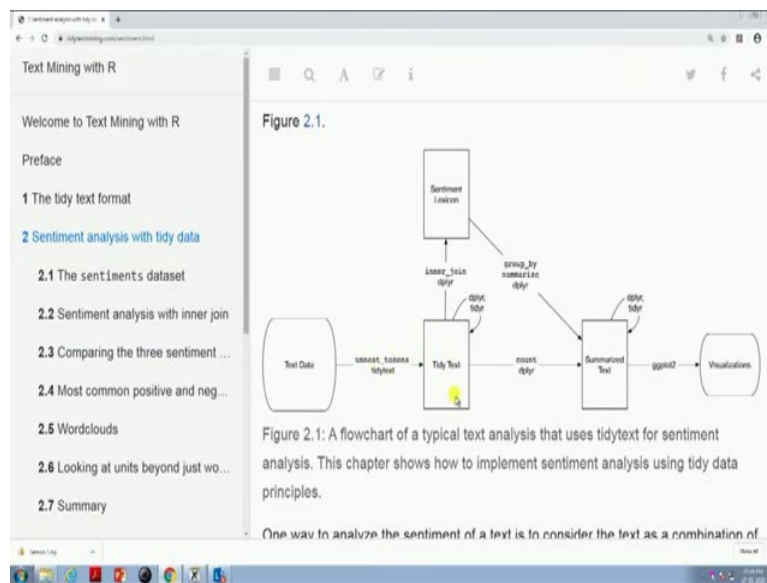
(Refer Slide Time: 04:01)



So welcome to text mining.R. Text mining with R is a book is a free book you can also buy from Amazon and etcetera, but his is a free particular web version of the book. Written by Julia Silge and David Robinson and O'REILLY is a one of the I would say major publisher online publisher in this context who publishes lot of books which is where we can use R or python for text mining.

So there is a book called good book for R which is also good book with graphics of R which are also good books to start with R programming this one is only focused on text mining with R.

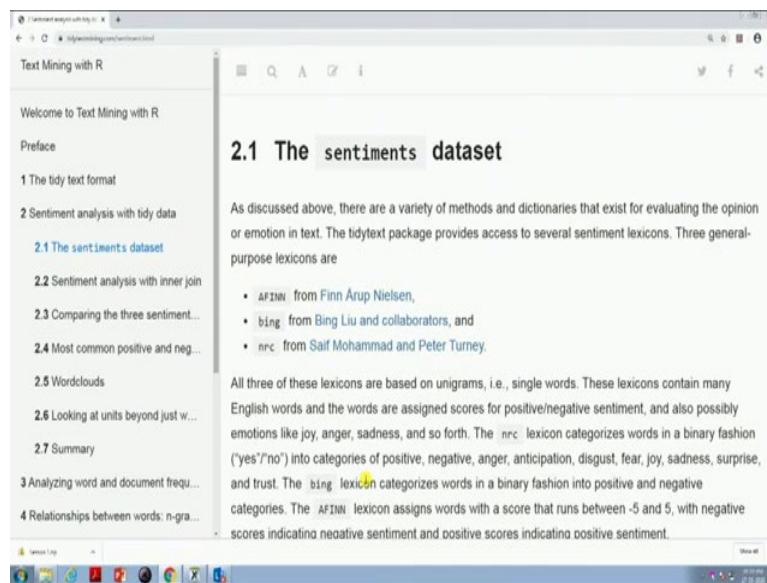
(Refer Slide Time: 04:49)



And we will be majorly focusing on this sentiment analysis with tidy text mining. So how to do this if you have a text data our first job this is the methodology that they have proposed. The first job that you do here if you can see this properly I will just make it a little bit bigger just check this picture. So the first job that you do is from the text data you come up with unnest token tidy text and you have created tidy text which is a clean version of the text.

Then with this clean version you join with a lexicon sentiment lexicon which is available lexicon means this libraries and you create the summarize text. Summarize text means text which were the, this thing the sentiment and etcetera is coded and with that with the g-plot library we will do certain kind of visualization. So in this particular session also we will be doing that.

(Refer Slide Time: 05:49)

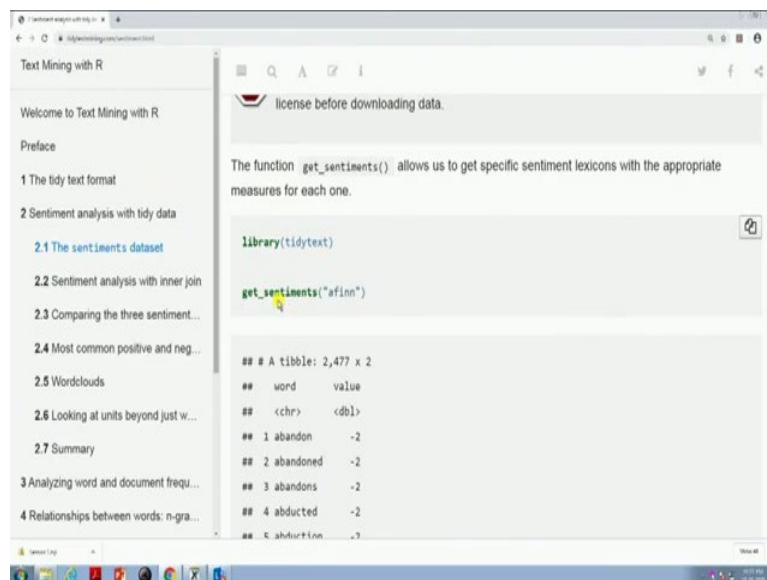


So there are 3 libraries that these guys are using if you carefully see the library names are Afinn the three library names are Afinn which is created by Finn Arup Nielsen and then there is Bing library which is another lexicon and there is NRC which is another which is another lexicon. So NRC is a lexicon where the, I would say if the definition is here so NRC lexicon categories words in binary fashion into categories positive, negative.

And also in various emotions like anger, anticipation, disgust, fear, joy, sadness, surprise and trust, so these are basically the core 8 I would say the basic 8 emotions that are there in the market so they break in that way also. And then if I focus on the rest of the part I see that bing lexicon which categories is into binary fashion only positive and negative probably least use lexicon.

And then there is Afinn which assigns words with a score between - 5 to + 5 as I was telling. There is - 5 to + 5 and which negative indicates negative sentiments and positive scores indicating positive sentiment. So we can use any of these three libraries my choice are generally Afinn or NRC. Afinn I do it for sentiment mining and NRC I do it for generally the emotion mining.

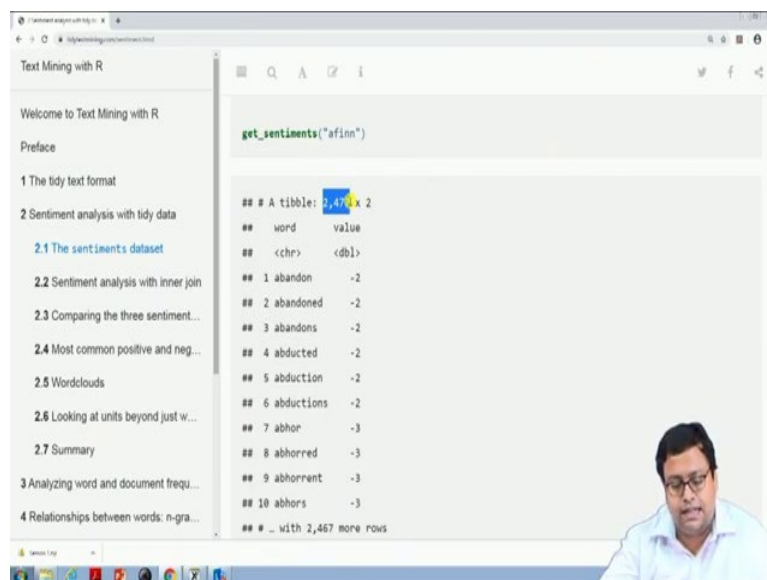
(Refer Slide Time: 07:26)



```
library(tidytext)

get_sentiments("afinn")

## # A tibble: 2,477 x 2
##   word      value
##   <chr>    <dbl>
## 1 abandon     -2
## 2 abandoned   -2
## 3 abandons    -2
## 4 abducted   -2
## 5 abduction   -2
```



```
get_sentiments("afinn")

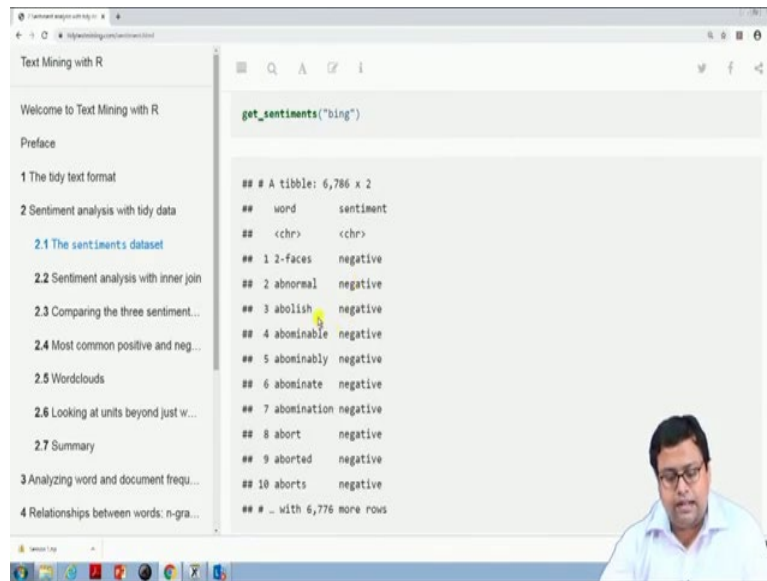
## # A tibble: 2,477 x 2
##   word      value
##   <chr>    <dbl>
## 1 abandon     -2
## 2 abandoned   -2
## 3 abandons    -2
## 4 abducted   -2
## 5 abduction   -2
## 6 abductions  -2
## 7 abhor       -3
## 8 abhorred    -3
## 9 abhorrent   -3
## 10 abhors     -3
## # ... with 2,467 more rows
```

So how to do that so let us say we have to call first the library called tidy text and now in this library after calling this library tidy text if you write get_sentiments and then Afinn so you can copy each of this codes by one by one paste it in your R programming this thing and you can see that what is happening. So if I just get _ sentiments and within that Afinn you will say that there are basically a table of 2477 words.

And there are two columns that means one is a word and one is the corresponding score. For Afinn, Afinn gives - 5 to + 5 if you remember. So here it is saying that abandon is a negative word abandoned is also negative - 2 and then let us say abhor is - 3, abhorrent is also - 3 so that kind of coding they have done and they have done for 2467 total 2477 words are there.

So some words are different form of the same word and correspondingly the sentiments are given.

(Refer Slide Time: 08:43)

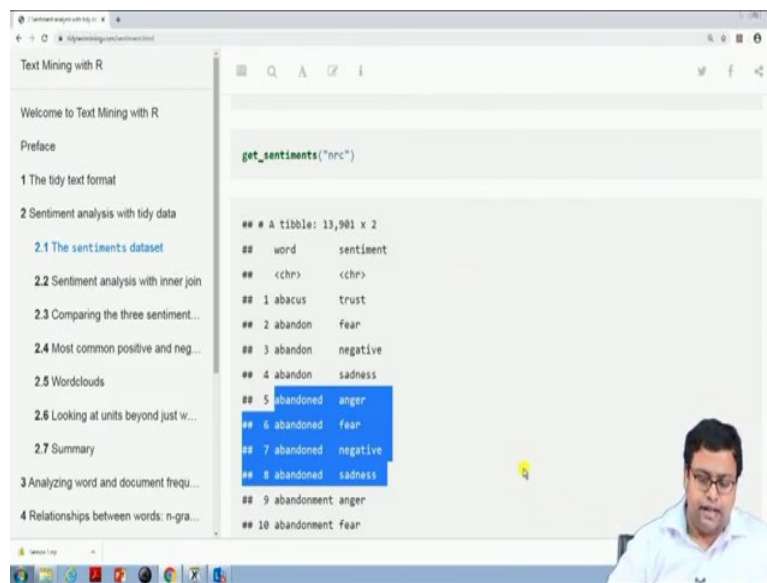


```
get_sentiments("bing")

## # A tibble: 6,786 x 2
##   word      sentiment
##   <chr>    <chr>
## 1 2-faces  negative
## 2 abnormal negative
## 3 abolish  negative
## 4 abominable negative
## 5 abominably negative
## 6 abominate negative
## 7 abomination negative
## 8 abort    negative
## 9 aborted  negative
## 10 aborts   negative
## # ... with 6,776 more rows
```

The same thing if I do it for Bing another library it only gives me negative or positive and the words are different. The words that this guy cover is more words 6786 words, but they only give positive or negative that is all. While the scope of the words are high the I would say the information content in this particular library is much lower than the previous one. So that might be a challenge.

(Refer Slide Time: 09:14)



```
get_sentiments("nrc")

## # A tibble: 13,901 x 2
##   word      sentiment
##   <chr>    <chr>
## 1 abacus   trust
## 2 abandon  fear
## 3 abandon  negative
## 4 abandon  sadness
## 5 abandoned anger
## 6 abandoned fear
## 7 abandoned negative
## 8 abandoned sadness
## 9 abandonment anger
## 10 abandonment fear
```

Then when I do it for NRC they are further bigger you see they are 13901 words so that is why NRC is one of the most used lexicon in marketing even in marketing research also and you will see that they have given various words corresponding sentiments. The problem here is that you see that abandoned one word might be probably associated with multiple sentiment like abandoned this particular thing is related with anger.

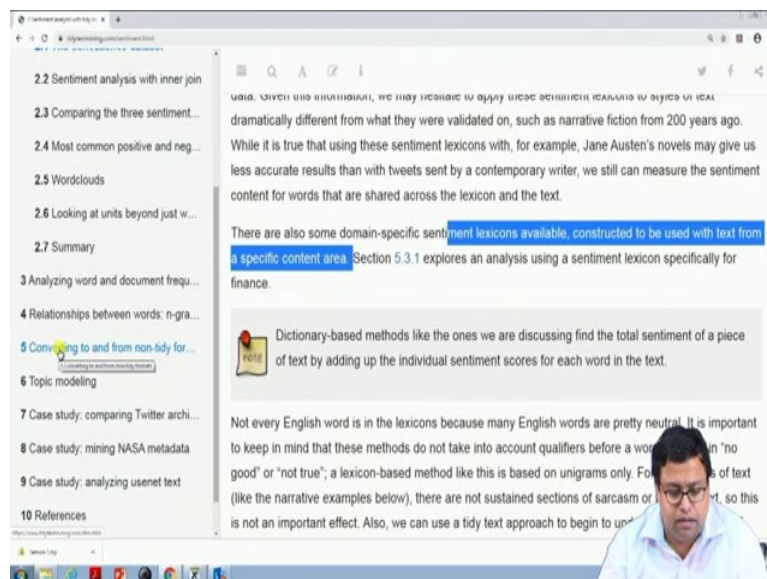
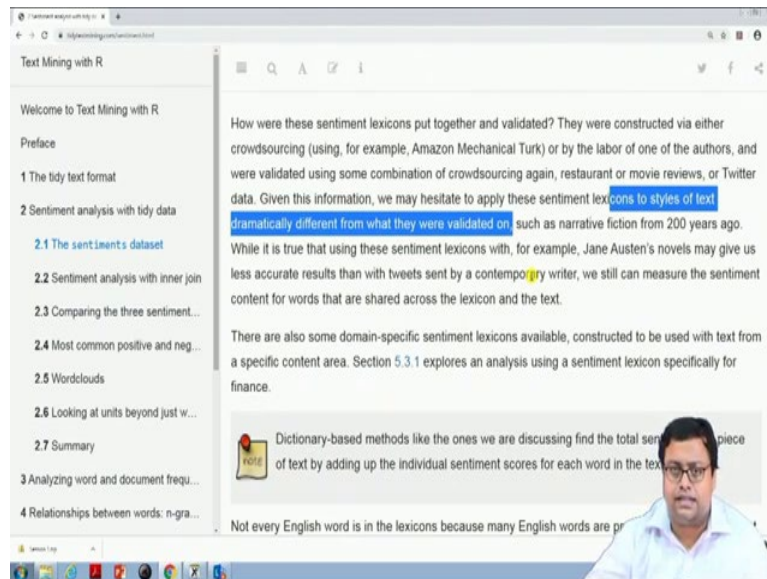
Related with fear so I am abandoned in an Island let us say so which is also associated with fear. So if somebody abandons you, you will be angry to him or if you are angry with somebody you can abandon him, but if you are abandoned then you will face fear and probably sadness also. So all of these things together is basically a negative sentiment. So there are sentiment rating also, sentiment classification also and there are emotion classifications also.

Now all of this classifications are not always correct, but they do they have been applied in multiple places, multiple contexts and based on that they have done this. So we can fairly believe on this system. So ideally what we generally do is we see that in how many different kind of papers, how many different type of places this lexicons are used and if you want to create your own lexicon in your own language you have to follow their footsteps so whatever they have done.

First of you all you have to create a Naive Bayes or any something based that kind of a scoring pattern and then you have to try out in multiple other-other options to fine tune the scoring and later at the end you will say that okay these words in different languages say in

Hindi language or in a Bengali language these words is related to positive, these words are related to negative you can create that.

(Refer Slide Time: 11:18)



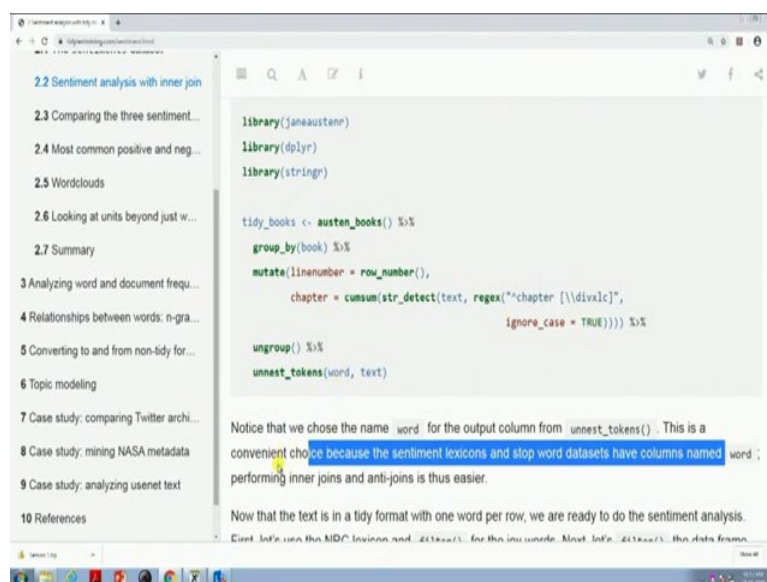
So how are these sentiments lexicons put together and validated? They were constructed via either crowdsourcing or by the labor of one of the authors and were validated using some combination of outsourcing again. Restaurant and movie reviews or Twitter data is something that is actually use. For example I use hotel data they also use this kind of restaurant or movie reviews data.

So we may hesitate to apply this sentiment lexicons to styles of text dramatically different from what they are validated on. Probably let us say how Indian people speak in English or

write English or how American people or western people write English might be very different. So it has to be used contextually you cannot use a lexicon which has been developed in a European context in Indian dataset sometimes that might have a back firing.

So that kind of thing you should keep in mind or at least if you are doing research you should keep that in mind and you should write about that limitation. There are some also some domain specific sentiment lexicons available constructed to be used with text with a specific content area. Section 5.3.1 later point of time you will be dealing with that. So right now I am not focusing on that.

(Refer Slide Time: 12:39)



```
library(janeaustenr)
library(dplyr)
library(stringr)

tidy_books <- austen_books() %>%
  group_by(book) %>%
  mutate(linenumber = row_number(),
         chapter = cumsum(str_detect(text, regex("^chapter [\\d\\v\\c]",
                                               ignore_case = TRUE)))) %>%
  ungroup() %>%
  unnest_tokens(word, text)
```

Notice that we chose the name `word` for the output column from `unnest_tokens()`. This is a convenient choice because the sentiment lexicons and stop word datasets have columns named `word`; performing inner joins and anti-joins is thus easier.

Now that the text is in a tidy format with one word per row, we are ready to do the sentiment analysis.

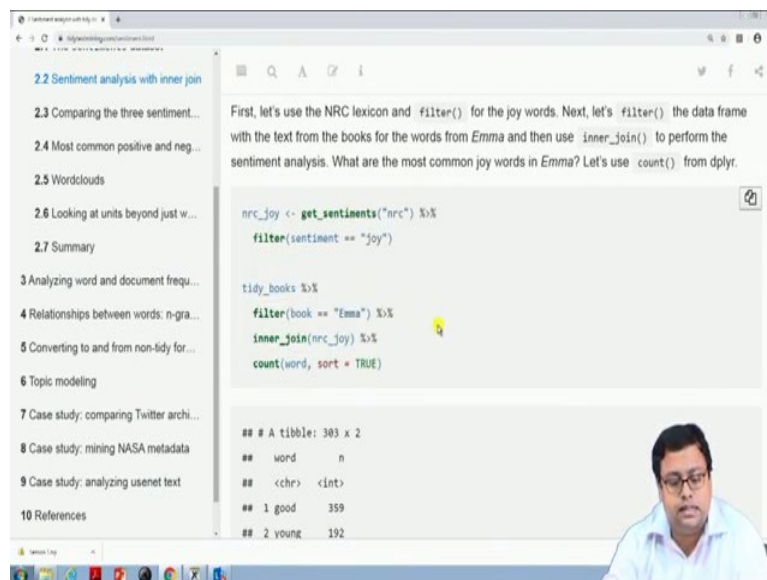
Next, let's use the MDC lexicon and `extract()` for the in-words. Most info. `extract()` the data frame

Now how to do, how to use this? So they are calling certain library called janeaustenr and then dplyr and stringr. So these are common libraries. This one is the library where the books are there austen books are there janeaustenr books. So then it is calling this tidy books tidy books is a dataset that you will be creating using austen_books. So books of austen which will be coming from this library.

And you will group by book. So the whole corpus you are grouping by books and then mutate how line number is equal to row number. So every this thing will have a line number then you are also find down the chapter number. So the dataset will have line number and chapter number and then again you are ungrouping the data and then you are changing trying to create from the text the tokens the words.

So notice that we choose the name word for the output columns from unnest tokens. This is a convenient choice because the sentiment lexicons and stop words datasets have column named words. So these things is same in both the cases performing inner joins and anti joins is thus easier. So we will see what they are doing. Now what they are doing, they are saying that get sentiment NRC and filter sentiment is equal to joy. So they are creating basically a dataset where only joy related words are there.

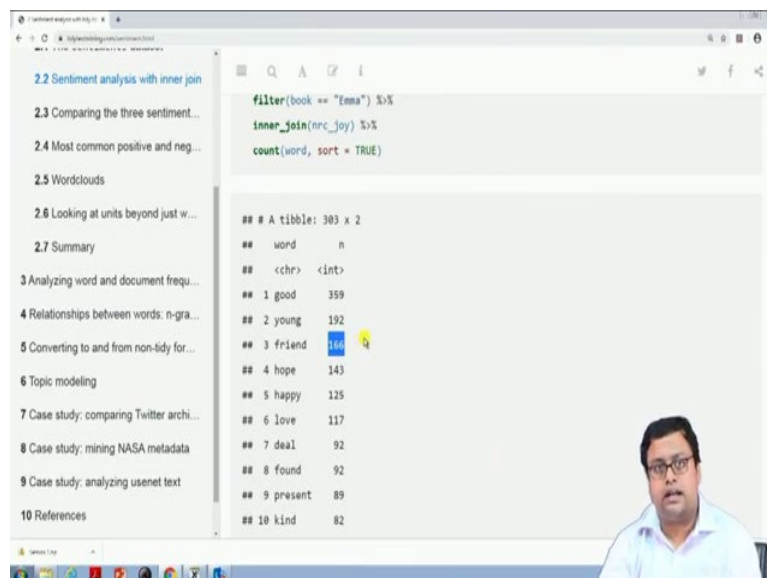
(Refer Slide Time: 14:18)



```
nrc_joy <- get_sentiments("nrc") %>%
  filter(sentiment == "joy")

tidy_books %>%
  filter(book == "Emma") %>%
  inner_join(nrc_joy) %>%
  count(word, sort = TRUE)
```

```
## # A tibble: 303 x 2
##   word      n
##   <chr> <int>
## 1 good    359
## 2 young   192
```



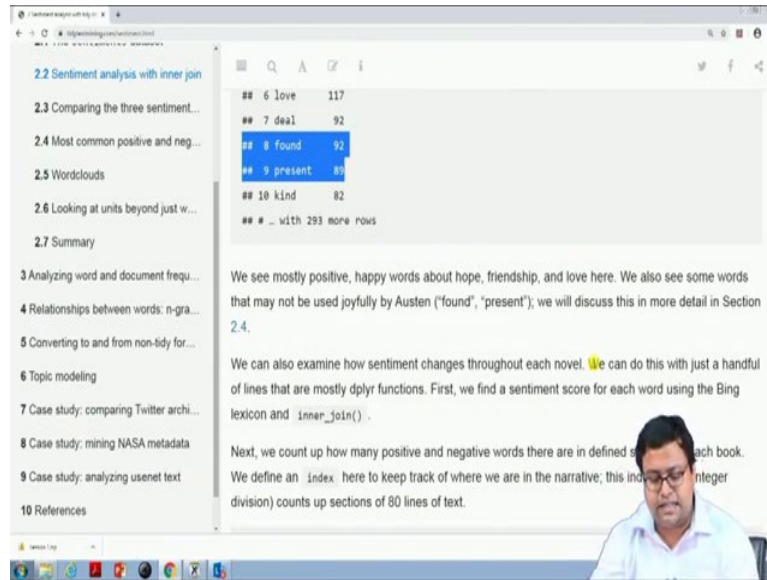
```
filter(book == "Emma") %>%
  inner_join(nrc_joy) %>%
  count(word, sort = TRUE)
```

```
## # A tibble: 303 x 2
##   word      n
##   <chr> <int>
## 1 good    359
## 2 young   192
## 3 friend   166
## 4 hope    143
## 5 happy   125
## 6 love    117
## 7 deal     92
## 8 found     92
## 9 present   89
## 10 kind     82
```

Now in this tidy books dataset that they have created they are finding out the Emma book the book name is Emma and in this book they are finding out which word is associated with joy only those words they are taking okay and they are trying to find out what is the count of those words. So now this good which is related to a joy is happening 359 times in the dataset.

Young which is also associated with let us say joy is 192 so I feel Young which is related to joy which has been occurring in this Emma book 192 times. Friend is occurring 166 times and so on. So that kind of data they are getting.

(Refer Slide Time: 15:01)



The screenshot shows a Jupyter Notebook interface. On the left is a table of contents with sections 2.2 through 10. The main area displays a code cell with the following output:

```
## 6 love 117
## 7 deal 92
## 8 found 92
## 9 present 80
## 10 kind 82
## ... with 293 more rows
```

The row for 'found' is highlighted in blue. Below the code cell, there is explanatory text:

We see mostly positive, happy words about hope, friendship, and love here. We also see some words that may not be used joyfully by Austen ("found", "present"); we will discuss this in more detail in Section 2.4.

We can also examine how sentiment changes throughout each novel. We can do this with just a handful of lines that are mostly dplyr functions. First, we find a sentiment score for each word using the Bing lexicon and `inner_join()`.

Next, we count up how many positive and negative words there are in defined sections of each book. We define an `index` here to keep track of where we are in the narrative; this `index` (integer division) counts up sections of 80 lines of text.

So we see mostly positive happy words about hope, friendship and love here. We also see some words that may not be joyful by Austen. For example found or present these words may not be present can have different meaning.

But found I do not know whether that is actually joy related or not. We can also examine how sentiment changes throughout the novel. So in the first part of the novel, second part of the novel how it is changing they try to check that.

(Refer Slide Time: 15:33)

The screenshot shows a presentation slide with a table of contents on the left and a main content area on the right. The table of contents includes items 2.2 through 2.7, 3, 4, 5, 6, 7, 8, 9, and 10. The main content area contains the following text and code:

Small sections of text may not have enough words in them to get a good estimate of sentiment while really large sections can wash out narrative structure. For these books, using 80 lines works well, but this can vary depending on individual texts, how long the lines were to start with, etc. We then use `spread()` so that we have negative and positive sentiment in separate columns, and lastly calculate a net sentiment (positive - negative).

```
library(tidyverse)

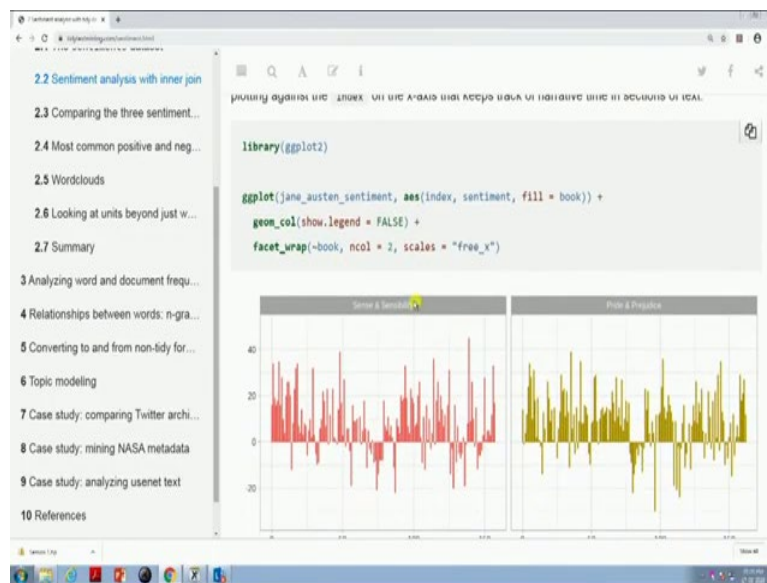
jane_austen_sentiment <- tidy_books %>%
  inner_join(get_sentiments("bing")) %>%
  count(book, index = linenumber %/% 80, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)
```

Now we can plot these sentiment scores across the plot trajectory of each novel plotting against the `index` on the x-axis that keeps track of narrative time in

So small section of text may not have enough words in them to get a good estimate of sentiment while large sections can wash out narrative structure. For these books using 80 lines works well. So every 80 line they are finding out the sentiment, but this can vary depending on the individual text how long the lines were to start with so that is why they are using this. So they are using tidy as the library and using Bing as the sentiment.

Bing if you remember derives positive or negative and then they try to find out that total sentiment how much it is positive - negative. So positive count every area breaking it into 80 and every 80 text they are finding out within that 80 set text how many times a positive word is happening, how many times the negative word is happening? Positive - negative is the sentiment that is how they are calculating and then they are putting it in a gg-plot.

(Refer Slide Time: 16:31)



So if I just see the book called sense and sensibility you will see that initially it was positive then it go into negative then again positive then again negative. So there is always a + and -, + and -, + and - and in fact you will see all the books Pride and Prejudice and then Mansfield Park or Emma, but Emma is mutually positive.

So here sense and sensibility or pride and prejudice is probably more I would say successful books than this two. Because there were lots of jumps up and down so here there was up then there is a down then ups then down. So sometimes there was some kind of pressure building up was much higher in this two cases then Emma, Emma is straightforward probably everything positive in the last there will be some negative, but majorly positive.

Here also in Mansfield Park straightforward positive major negativity is in the last. And then from a negative sentiment all of a sudden to very positive sentiment which might not be a good way. Again Northanger Abbey and persuasion, persuasion is absolutely positive everything is positive so that might be.

(Refer Slide Time: 17:47)



So we can see in figure 2.2 how the plot of each novel changes towards more positive and negative sentiment over the trajectory. Now these things this kind of an idea that how much is positive how much is negative. How many switches are there can be used to predict whether a book will be successful or not. Let us see you have a dataset of lots of books and then you know that whether this books were successful or not so that means you are y variable whether the books were successful or not.

And then you have the books text also and from there you can find out that how much positivity is there, how much different kind of sentiments or emotions are there. And whether they were positive or negative and how many changes have happened, all of these things can predict a book success.

Now a new writer probably comes with some kind of manuscript you can put it in your algorithm and can predict the success probable success obviously after controlling for who is the author, who is the controlling for all this stuff that is a basic recipe you can get that how to write a more successful book. So that kind of things are coming up in the coming days.

(Refer Slide Time: 19:03)

2.2 Sentiment analysis with inner join
2.3 Comparing the three sentiment...
2.4 Most common positive and neg...
2.5 Wordclouds
2.6 Looking at units beyond just w...
2.7 Summary
3 Analyzing word and document frequ...
4 Relationships between words: n-gra...
5 Converting to and from non-tdy for...
6 Topic modeling
7 Case study: comparing Twitter archi...
8 Case study: mining NASA metadata
9 Case study: analyzing usenet text
10 References

2.3 Comparing the three sentiment dictionaries

With several options for sentiment lexicons, you might want some more information on which one is appropriate for your purposes. Let's use all three sentiment lexicons and examine how the sentiment changes across the narrative arc of *Pride and Prejudice*. First, let's use `filter()` to choose only the words from the one novel we are interested in.

```
pride_prejudice <- tidy_books %>%  
  filter(book == "Pride & Prejudice")  
  
pride_prejudice
```

```
## # A tibble: 122,284 x 4  
##   book      linenumber chapter word
```

A video inset shows a man with glasses and a light blue shirt looking at the screen.

changes across the narrative arc of *Pride and Prejudice*. First, let's use `filter()` to choose only the words from the one novel we are interested in.

```
pride_prejudice <- tidy_books %>%  
  filter(book == "Pride & Prejudice")  
  
pride_prejudice
```

```
## # A tibble: 122,284 x 4  
##   book      linenumber chapter word  
##   <fct>          <int> <int> <chr>  
## 1 Pride & Prejudice      1     0 pride  
## 2 Pride & Prejudice      1     0 and  
## 3 Pride & Prejudice      1     0 prejudice  
## 4 Pride & Prejudice      3     0 by  
## 5 Pride & Prejudice      3     0 jane  
## 6 Pride & Prejudice      3     0 austen
```

A video inset shows the same man from the previous slide.


```

summarise(sentiment = sum(value)) %>%
mutate(method = "AFINN")

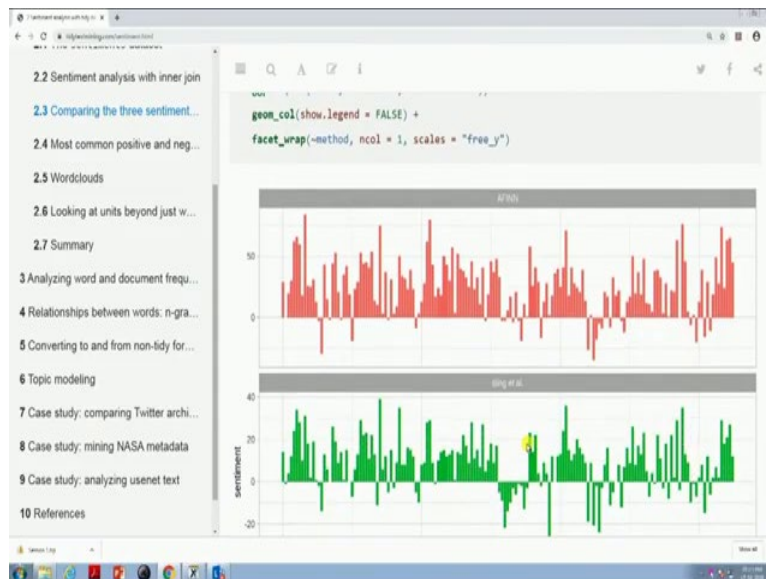
bing_and_nrc <- bind_rows(pride_prejudice %>%
  inner_join(get_sentiments("bing")) %>%
  mutate(method = "Bing et al."),
  pride_prejudice %>%
  inner_join(get_sentiments("nrc")) %>%
  filter(sentiment %in% c("positive",
    "negative"))) %>%
  mutate(method = "NRC") %>%
count(method, index = lineNumber %/% 80, sentiment) %>%
spread(sentiment, n, fill = 0) %>%
mutate(sentiment = positive - negative)

```

We now have an estimate of the net sentiment (positive - negative) in each chunk of text for each sentiment lexicon. Let's bind them together and visualize them in Figure 2.3.

So comparing the 3 sentiment dictionary so in this section 2.3 they have compared the three sentiments like for pride and prejudice this is the dataset that they have used. First they have used the AFINN library and then they have used the Bing library and then they have used and they counted in all of this things how much was that.

(Refer Slide Time: 19:33)

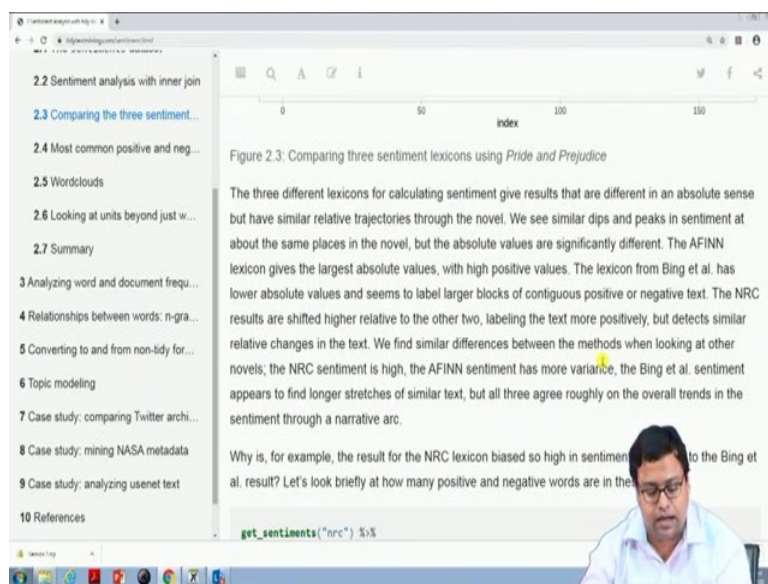




And if I just so this is basically Afinn and this is Bing if I just compare Afinn, Bing and NRC. NRC is giving heavily positive everything is positive. On the other hand Afinn and Bing are probably be able to show similar results. So Afinn and Bing are more close to each other and NRC is not so close to each other not so close to Bing. So it is probably and that can be a reason which we can say that I can shift the scale.

So if I compare this two I can say that if instead of this NRC 5 or - 10 if I do then this two might match with each other. So that we will see.

(Refer Slide Time: 20:20)



So there are three different lexicons for calculating sentiment give result they are different in an absolute sense, but similar in relative trajectories through the novel. We see similar dips

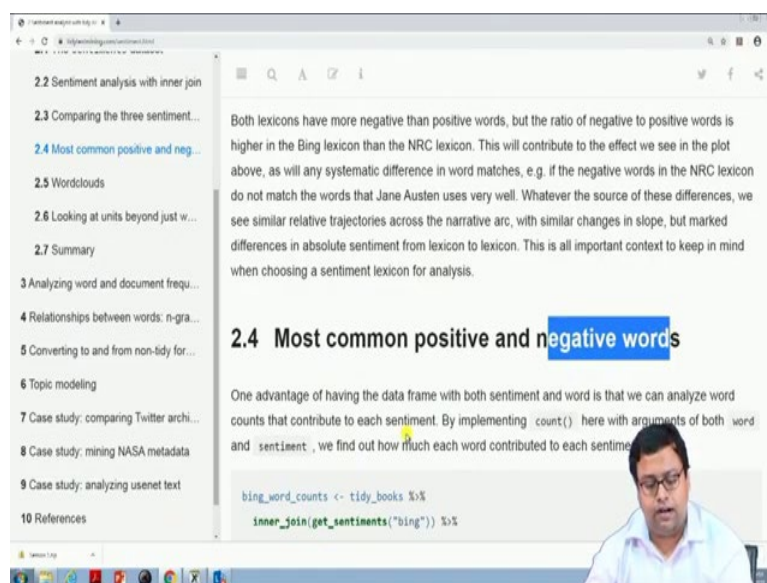
and peaks in sentiment at about the same places in the novel, but absolute values are significantly different.

The Affin lexicon gives the largest absolute values high positive values and lexicon Bing et al has lower absolute values and seems to be label larger blocks of contagious positive and negative text. So this is something that is important that not always you can take these libraries whatever their sentiment saying that is the sentiment. You have take that relative scale and because these are scales their measurements are different.

So in the relative scale you can say that okay this one is this part of the book is more sentimental than that part of book, but that value 3 sentiment value 3 or sentiment value 5 or sentiment value 50 exact absolute values has no meaning because what is sentiment high sentiment to him might not be a high sentiment to somebody else. Some people are more expressive, some people are less expressive.

So the people who have created these libraries can also be different type of people. So sometimes that thing you have to keep in mind that it is only the relative scale in which you have to measure this or read this not in an absolute scale. Then what can I do?

(Refer Slide Time: 21:51)



The image shows a presentation slide with a table of contents on the left and a main content area on the right. The table of contents includes items 2.2 through 2.7, 3, 4, 5, 6, 7, 8, 9, and 10. The main content area is titled '2.4 Most common positive and negative words' and contains text explaining the difference between the Bing and NRC lexicons. It also includes a code block with R code for sentiment analysis.

2.2 Sentiment analysis with inner join
2.3 Comparing the three sentiment...
2.4 Most common positive and neg...
2.5 Wordclouds
2.6 Looking at units beyond just w...
2.7 Summary
3 Analyzing word and document frequ...
4 Relationships between words: n-gra...
5 Converting to and from non-lyd for...
6 Topic modeling
7 Case study: comparing Twitter archi...
8 Case study: mining NASA metadata
9 Case study: analyzing usenet text
10 References

Both lexicons have more negative than positive words, but the ratio of negative to positive words is higher in the Bing lexicon than the NRC lexicon. This will contribute to the effect we see in the plot above, as will any systematic difference in word matches, e.g. if the negative words in the NRC lexicon do not match the words that Jane Austen uses very well. Whatever the source of these differences, we see similar relative trajectories across the narrative arc, with similar changes in slope, but marked differences in absolute sentiment from lexicon to lexicon. This is all important context to keep in mind when choosing a sentiment lexicon for analysis.

2.4 Most common positive and negative words

One advantage of having the data frame with both sentiment and word is that we can analyze word counts that contribute to each sentiment. By implementing `count()` here with arguments of both `word` and `sentiment`, we find out how much each word contributed to each sentiment.

```
bing_word_counts <- tidy_books %>%  
  inner_join(get_sentiments("bing")) %>%
```

```
bing_word_counts <- tidy_books %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()

bing_word_counts
```

```
## # A tibble: 2,585 x 3
##   word      sentiment     n
##   <chr>    <chr>    <int>
## 1 miss     negative  1855
## 2 well     positive  1523
## 3 good     positive  1380
## 4 great    positive   981
## 5 like     positive   725
## 6 better   positive   639
```

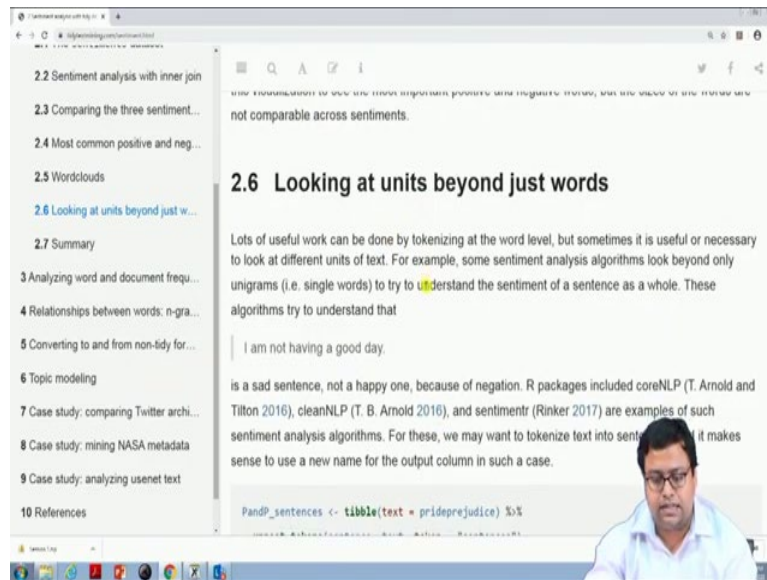
```
facet_wrap(~sentiment, scales = 'free_y') +
  labs(y = "Contribution to sentiment",
       x = NULL) +
  coord_flip()
```

The visualization consists of two horizontal bar charts. The left chart, titled 'negative', shows the most common negative words: miss, poor, doubt, object, sorry, impossible, afraid, scarcely, bad, and anxious. The right chart, titled 'positive', shows the most common positive words: well, good, great, like, better, enough, happy, love, pleasure, and happiness.

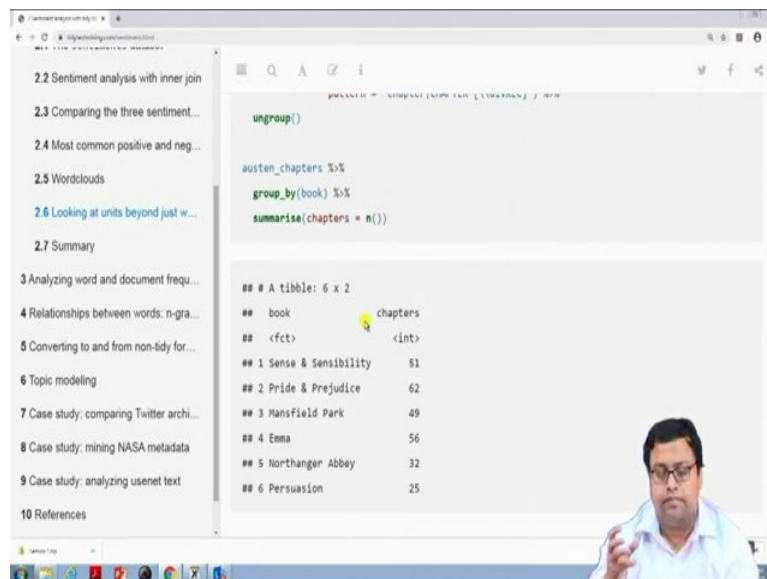
What are the most common positive and negative words? So they have used using Bing that is the most common and they have found out that in this dataset the dataset is in this case the dataset is basically. So they have checked for all of them and they found that miss, poor, doubt these are the most common afraid, impossible, sorry these are most common negative words. And well, good, great, better, enough, happy these are the most common negative words.

So I will not do that because these are something that you should do on your own now. We have taken a path of 11 weeks. So now it is time that you can get resources from online on your own and find out that how I can use that in my training.

(Refer Slide Time: 23:24)



The screenshot shows a presentation slide with a table of contents on the left and a main content area on the right. The main content area is titled "2.6 Looking at units beyond just words" and discusses tokenizing text at the sentence level. It includes an example sentence: "I am not having a good day." and explains that this is a sad sentence due to negation. It mentions R packages like coreNLP, cleanNLP, and sentimentr. At the bottom, there is a code snippet: `PandP_sentences <- tibble(text = prideprejudice) %>%`. A small video inset of a man is visible in the bottom right corner.



The screenshot shows a presentation slide with a table of contents on the left and a main content area on the right. The main content area displays R code for summarizing chapters by book. The code includes `ungroup()`, `austen_chapters %>%`, `group_by(book) %>%`, and `summarise(chapters = n())`. The output is a tibble with 6 rows and 2 columns: 'book' and 'chapters'. The output is as follows:

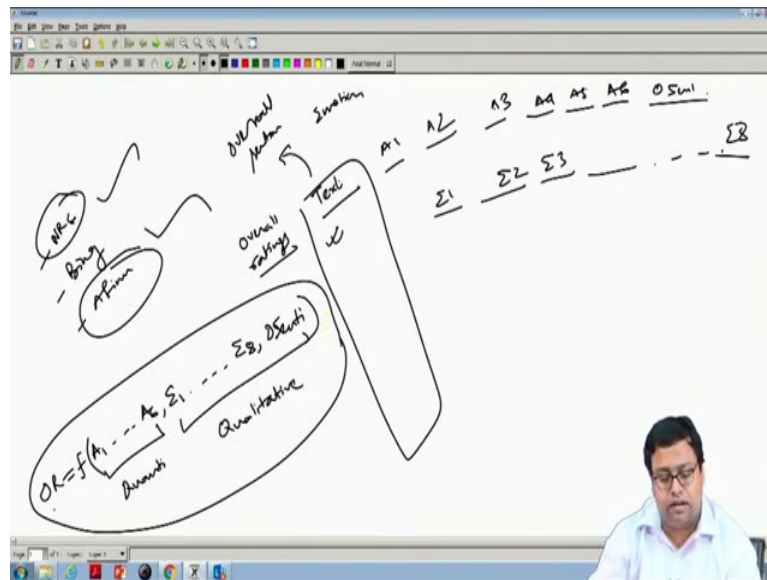
book	chapters
1 Sense & Sensibility	51
2 Pride & Prejudice	62
3 Mansfield Park	49
4 Emma	56
5 Northanger Abbey	32
6 Persuasion	25

A small video inset of a man is visible in the bottom right corner.

And then we are looking at units beyond just words we can also find out however little known so in this particular thing they are also thinking about that how instead of words I can take more let us say good is positive.

But if I say not good that is negative actually. So it might give absolutely opposite setting so how I can look beyond words when I am doing data science I am doing sentiment mining.

(Refer Slide Time: 23:55)



So in using these 3 things basically NRC, Bing and Afinn and probably majorly NRC and Afinn we will be doing some work in the next video. If you remember we have overall rating of hotels.

I have some text and I have 6 aspects A1 which is location, food, this, this, this and this. So can I find out from this text, can I find out the overall sentiment of this text and emotions in this text. So I can put from this text I will put this NRC and Afinn library and I can create another column which is overall sentiment and then 8 emotions let us say E1, E2, E3, ..., E8, 8 emotions.

Now I can say overall rating is a function of obviously the aspects A1 to A6, but it is also function of how and which emotion you are in and what is your overall sentiment. So not only quantitative rating is impacting your overall rating, but also qualitative rating, qualitative aspects are also impact your overall rating. So can we create how the qualitative part gives enough information for a manager to see that how that impacts the customer ratings that they are giving we will be trying to see.

So this particular tidy text whatever we are doing here this course we will be using to solve this problem in the next video. So just I would before you start the next video, I would strongly suggest that is why that only the text mining part you can if somebody can learn the whole thing that is best.

(Refer Slide Time: 26:00)

book	chapter	negativewords	words	ratio
1 Sense & Sensibility	43	161	3405	0.0473
2 Pride & Prejudice	34	111	2104	0.0528
3 Mansfield Park	46	173	3685	0.0469
4 Emma	15	151	3340	0.0452
5 Northanger Abbey	21	149	2982	0.0500
6 Persuasion	4	62	1807	0.0343

These are the chapters with the most sad words in each book, normalized for number of words in the chapter. What is happening in these chapters? In Chapter 43 of *Sense and Sensibility* Mr. Ferris is seriously ill, near death, and in Chapter 34 of *Pride and Prejudice* Mr. Darcy proposes for the first time (so badly!). Chapter 46 of *Mansfield Park* is almost the end, when everyone learns of the scandalous adultery. Chapter 15 of *Emma* is when horrifying Mr. Elton proposes, and in Chapter 21 of *Northanger Abbey* Catherine is deep in her Gothic faux fantasy of murder, etc.

But this chapter 2 you should do it thoroughly, you should read the chapter 2 thoroughly. Pick up the codes run it in your run and see that whether you are getting the results if you are getting stuck you can email me or message me. So that is all thank you for being in this particular video and I will see you in the next video. Thanks.