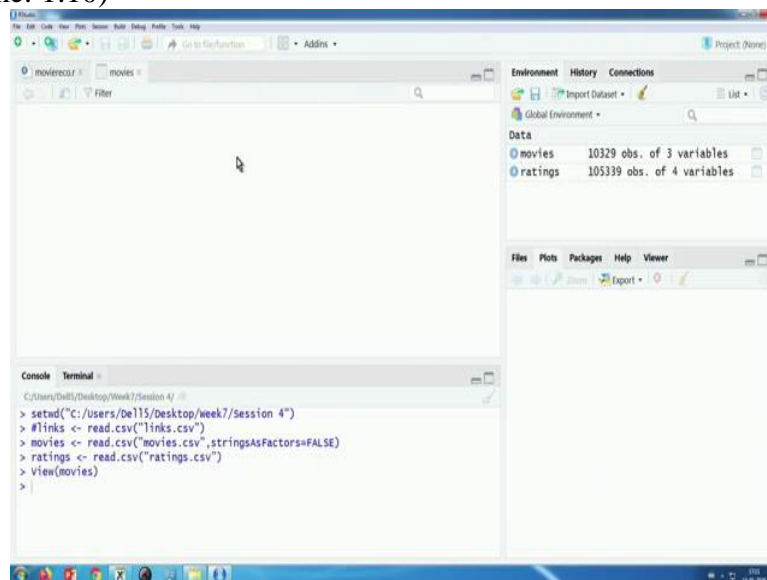


Marketing Analytics
Prof. Swagato Chatterjee
Vinod Gupta School of Management
Indian Institute of Technology, Kharagpur
Lecture 38: Recommendation Engine and Retail Analytics (Contd.)

Hello, everybody, welcome to Marketing Analytics Course, this is Dr. Swagato Chatterjee from the VGSOM IIT, Kharagpur who is taking this course. We are in week seven, and we are discussing Recommendation Engine. So, in this video and probably in the next video also, we will discuss certain examples of recommendation engine and their applications with real world data.

So, in this particular data set, here, we have movie ratings and where people have given ratings for movies. And I also have genre of the movies and I want to find out how I can recommend certain movies based on this genre and based on this, I would say interest towards those genre where data is being collected from let us say, IMDb or let us say Netflix or something like that.

(Refer Slide Time: 1:10)



The screenshot shows the RStudio interface. The console window at the bottom contains the following R code:

```
C:\Users\De115\Desktop\Week7\Session 4> #  
> setwd("C:/Users/De115/Desktop/week7/Session 4")  
> #links <- read.csv("links.csv")  
> movies <- read.csv("movies.csv", stringsAsFactors=FALSE)  
> ratings <- read.csv("ratings.csv")  
> view(movies)  
> |
```

The Environment pane on the right shows two data objects: 'movies' with 10329 observations and 3 variables, and 'ratings' with 105339 observations and 4 variables.

Environment History Connections

Global Environment

Data

- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables

Files Plots Packages Help Viewer

Console Terminal

```
C:/Users/Dell/Desktop/Week7/Session 4/ >
> setwd("c:/users/dell/desktop/week7/session 4")
> #links <- read.csv("links.csv")
> movies <- read.csv("movies.csv", stringsAsFactors=FALSE)
> ratings <- read.csv("ratings.csv")
> View(movies)
> View(ratings)
```

movieid	title	genres
1	Toy Story (1995)	Adventure(Animation Children Comedy Fantasy)
2	Jurassic (1995)	Adventure(Children Fantasy)
3	Grumpier Old Men (1995)	Comedy Romance
4	Waiting to Exhale (1995)	Comedy Drama Romance
5	Father of the Bride Part II (1995)	Comedy
6	Heat (1995)	Action Crime Thriller
7	Subrina (1995)	Comedy Romance
8	Tom and Huck (1995)	Adventure Children
9	Sudden Death (1995)	Action
10	GoldenEye (1995)	Action Adventure Thriller

Environment History Connections

Global Environment

Data

- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables

Files Plots Packages Help Viewer

Console Terminal

```
C:/Users/Dell/Desktop/Week7/Session 4/ >
> setwd("c:/users/dell/desktop/week7/session 4")
> #links <- read.csv("links.csv")
> movies <- read.csv("movies.csv", stringsAsFactors=FALSE)
> ratings <- read.csv("ratings.csv")
> View(movies)
> View(ratings)
> unique(ratings$userId)
```

userId	movieid	rating	timestamp
1	1	16	4.0 1217897793
2	1	24	1.5 1217895807
3	1	32	4.0 1217896246
4	1	47	4.0 1217896556
5	1	50	4.0 1217896523
6	1	110	4.0 1217896150
7	1	150	3.0 1217895940
8	1	161	4.0 1217897864
9	1	165	3.0 1217897135
10	1	204	0.5 1217895786

Environment History Connections

Global Environment

Data

- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables

Files Plots Packages Help Viewer

Console Terminal

```
C:/Users/Dell/Desktop/Week7/Session 4/ >
[523] 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540
[541] 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558
[559] 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576
[577] 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594
[595] 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612
[613] 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630
[631] 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648
[649] 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666
[667] 667 668
> |
```


The screenshot shows the RStudio interface with an R script being executed. An 'Install Packages' dialog box is open, showing the installation of 'recommenderlab' and 'ggplot2' from the CRAN repository. The console output includes the following R code and messages:

```

1 #links <- read.csv("links.csv")
2 movies <- read.csv("movies.csv", stringsAsFactors=FALSE)
3 ratings <- read.csv("ratings.csv")
4 #tags <- read.csv("tags.csv")
5
6 library(recommenderlab)
7 library(ggplot2)
8
9 ## Data pre-processing
10 genres <- as.data.frame(movies$genres)
11 library(data.table)
12 genres2 <- as.data.frame(tstrsplit(genres, '|'))
13
14 stringsAsFactors=FALSE
15 colnames(genres2) <- c(1:10)
16
17 #####
18
19 (Dp Level) :
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

Warning messages:
1: package 'recommenderlab' was built under R version 3.6.2
2: package 'arules' was built under R version 3.6.2
3: package 'proxy' was built under R version 3.6.2
> library(ggplot2)
Need help getting started? Try the cookbook for R:
<http://www.cookbook-r.com/Graphs/>
warning message:
package 'ggplot2' was built under R version 3.6.2
>

The screenshot shows the RStudio interface with the 'genres' data table loaded. The environment pane shows the following data:

Object	Class	Attributes
genres	data.table	10329 obs. of 1 variable
movies	data.table	10329 obs. of 3 variables
ratings	data.table	105339 obs. of 4 variables

The console shows the following R code and output:

```

> ## Data pre-processing
> genres <- as.data.frame(movies$genres, stringsAsFactors=FALSE)
> view(genres)
>

```

The screenshot shows the RStudio interface with the 'data.table' package being installed. The console output includes the following R code and messages:

```

1 #links <- read.csv("links.csv")
2 movies <- read.csv("movies.csv", stringsAsFactors=FALSE)
3 ratings <- read.csv("ratings.csv")
4 #tags <- read.csv("tags.csv")
5
6 library(recommenderlab)
7 library(ggplot2)
8
9 ## Data pre-processing
10 genres <- as.data.frame(movies$genres, stringsAsFactors=FALSE)
11 library(data.table)
12 genres2 <- as.data.frame(tstrsplit(genres[,1], '|'),
13                               type.convert=TRUE,
14                               stringsAsFactors=FALSE)
15 colnames(genres2) <- c(1:10)
16
17 #####
18
19 (Dp Level) :
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

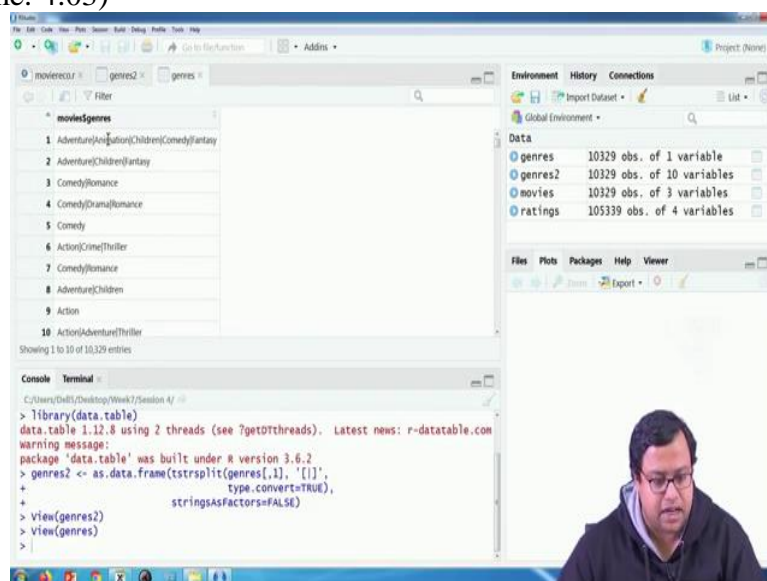
Warning messages:
package 'data.table' was built under R version 3.6.2
>

Now, if somebody has already rated a movie, then you will not see the movie anymore, then I have to find out his similar movie which he might like and likeness data, whether he will like or dislike a data, a movie will come from basically the rating that this guy has provided. So, I will ask for a library called recommender lab and ggplot2, recommender lab whatever we did before, the item based and user based collaborative filtering, this guy will do it on its own quickly.

For bigger data, it helps so, recommender lab and ggplot2 is two libraries that I am calling, if you have not installed it, you have to first install it and then and how to install it? You can write `install.packages` and then the package name or you can go here and install and write the package name here and then that will install the package. So, I have installed the package beforehand and I am just running this thing here. So, the first thing is processing of the data, I will break the data into genres.

So, for that what I am doing is I am creating a genres column, so which is a genre column, only the genres of my dataset and then I am using a library called `data.table`. And then in with this library, I am breaking the genre with `c`, there is called string split. So, this string split will actually split the screen based of the genre , 1, that means genre's first column, based on this particular sign. Whenever this sign occurs, it will just split it and then `stringAsFactors` is equal to false and it will create a dataset.

(Refer Slide Time: 4:03)



Environment History Connections

Global Environment

Data

- genres 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables

```

> library(data.table)
data.table 1.12.8 using 2 threads (see ?getDTthreads). Latest news: r-datatable.com
warning message:
package 'data.table' was built under R version 3.6.2
> genres2 <- as.data.frame(tstrsplit(genres[1,], '[ ]',
+                               type.convert=TRUE),
+                          stringsAsFactors=FALSE)
> view(genres2)
> view(genres)
  
```

Environment History Connections

Global Environment

Data

- genres 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables

```

> colnames(genres2) <- c(1:10)
> view(genres2)
  
```

```

9 ## data pre-processing
10 genres <- as.data.frame(movies$genres, stringsAsFactors=FALSE)
11 library(data.table)
12 genres2 <- as.data.frame(tstrsplit(genres[1,], '[ ]',
13                               type.convert=TRUE),
14                          stringsAsFactors=FALSE)
15 colnames(genres2) <- c(1:10)
16
17 genre_list <- c("Action", "Adventure", "Animation", "Children",
18              "Comedy", "Crime", "Documentary", "Drama", "Fantasy",
19              "Film-Noir", "Horror", "Musical", "Mystery", "Romance",
20              "Sci-Fi", "Thriller", "War", "Western") # we have 18 genres
21
22 genre_matrix <- matrix(0,10330,18) #empty matrix, 10330=no of movies+1, 18=no
23 genre_matrix[1,] <- genre_list #set first row to genre list
24 colnames(genre_matrix) <- genre_list #set column names to genre list
25
  
```

Environment History Connections

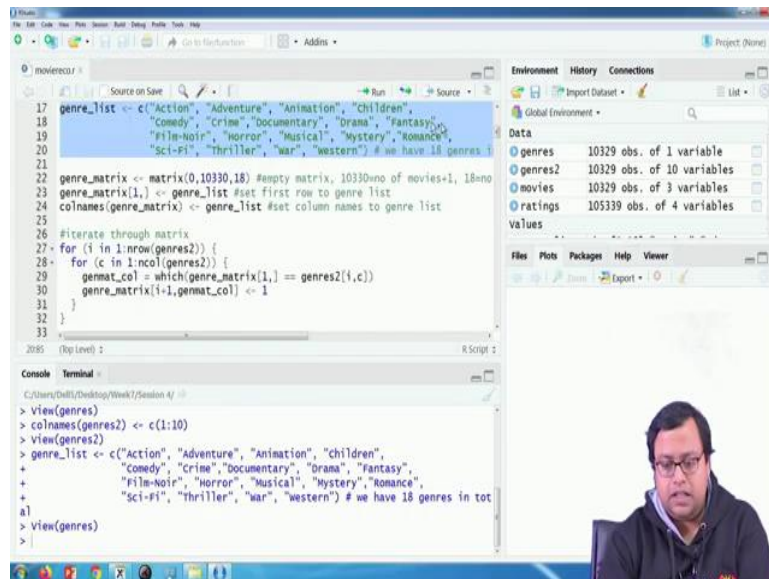
Global Environment

Data

- genres 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables

```

> library(data.table)
data.table 1.12.8 using 2 threads (see ?getDTthreads). Latest news: r-datatable.com
warning message:
package 'data.table' was built under R version 3.6.2
> genres2 <- as.data.frame(tstrsplit(genres[1,], '[ ]',
+                               type.convert=TRUE),
+                          stringsAsFactors=FALSE)
> view(genres2)
> view(genres)
  
```



So, genre2 gets, what it does? It is actually breaking if you check genres if it is adventure, animation, children, comedy and fantasy, so adventure, animation, children, comedy and fantasy so, that is how it has broken and the rest of them are blind. So, we have created and if I just column names of genres 2 if I make it 1 to 10 now, it will be clearly seen that I am actually breaking the genres of each column into separate separate columns. So, these are my various genres.

And if I will find out the unique of these genres in the whatever they are in this 1000, 3000, 10,329 observations of 10 variables, you will basically get 668 sorry, 18 different kinds of genres. So, these are 18 different kind of genres that is coming up. So, that is a genre list that we have and based on this genre list, I will create a matching so, whether what is the code for this particular genre?

(Refer Slide Time: 5:07)

Environment History Connections

Global Environment

Data

- genres 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables

Files Plots Packages Help Viewer

```
Console Terminal
C:/Users/Dalit/Desktop/Week7/Session 47
> colnames(genres2) <- c(1:10)
> view(genres2)
> genre_list <- c("Action", "Adventure", "Animation", "children",
+ "Comedy", "Crime", "Documentary", "Drama", "Fantasy",
+ "Film-Noir", "Horror", "Musical", "Mystery", "Romance",
+ "Sci-Fi", "Thriller", "War", "Western") # we have 18 genres in tot
a)
> view(genres)
> view(genres2)
>
```

Environment History Connections

Global Environment

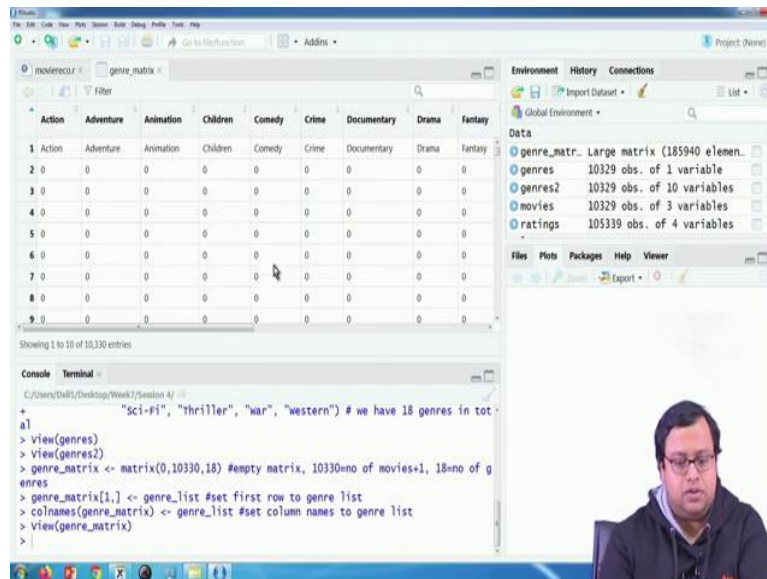
Data

- genre_matr. Large matrix (185940 elemen)
- genres 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables

Files Plots Packages Help Viewer

```
17 genre_list <- c("Action", "Adventure", "Animation", "children",
18 "Comedy", "Crime", "Documentary", "Drama", "Fantasy",
19 "Film-Noir", "Horror", "Musical", "Mystery", "Romance",
20 "Sci-Fi", "Thriller", "War", "Western") # we have 18 genres in tot
21
22 genre_matrix <- matrix(0,10330,18) #empty matrix, 10330no of movies+1, 18no of g
23 genre_matrix[1,] <- genre_list #set first row to genre list
24 colnames(genre_matrix) <- genre_list #set column names to genre list
25
26 Iterate through matrix
27 For (i in 1:nrow(genres2)) {
28   for (c in 1:ncol(genres2)) {
29     genmat_col = which(genre_matrix[1,] == genres2[i,c])
30     genre_matrix[i+1,genmat_col] <- 1
31   }
32 }
33
```

```
Console Terminal
C:/Users/Dalit/Desktop/Week7/Session 47
+ "Film-Noir", "Horror", "Musical", "Mystery", "Romance",
+ "Sci-Fi", "Thriller", "War", "Western") # we have 18 genres in tot
a)
> view(genres)
> view(genres2)
> genre_matrix <- matrix(0,10330,18) #empty matrix, 10330no of movies+1, 18no of g
enres
> genre_matrix[1,] <- genre_list #set first row to genre list
> colnames(genre_matrix) <- genre_list #set column names to genre list
>
```

So, if this is the first particular movies genre, so then I will get a matrix in such a way such that that matrix will have number of movies is 10,329. So 10,329 rows and 18 columns, 18 columns each has one genre. And the value will be zero if that particular movie falls in that genre, and that data will come from this particular matrix. And if it is not in that genre then 0, if it is in that genre then 1, so, that is what I am going to create now.

So what I am doing is that I am creating see, there is a matrix of 0 with 10,330, that means first row will be the column name and 18 columns. So that is what I am creating, the first row genre matrix 1, the first row is basically the genre list, the column names and then column names of genre matrix is basically the genre list so, that is what I am doing here. So, if you check the genre matrix right now, it is basically all zeros, okay. And then what I am doing is for $i = 1$ to n row of genres2 that means 10,330 and then $c = 1$ to n column of genres2, that means 1 to 18, what will I do? If it is matching then 1 otherwise 0.

(Refer Slide Time: 6:35)

```
21
22 genre_matrix <- matrix(0,10330,18) #empty matrix, 10330=no of movies+1, 18=no
23 genre_matrix[1,] <- genre_list #set first row to genre list
24 colnames(genre_matrix) <- genre_list #set column names to genre list
25
26 #iterate through matrix
27 for (i in 1:nrow(genres2)) {
28   for (c in 1:ncol(genres2)) {
29     genmat_col = which(genre_matrix[1,] == genres2[i,c])
30     genre_matrix[i+1,genmat_col] <- 1
31   }
32 }
33
34 #convert into dataframe
35 genre_matrix2 <- as.data.frame(genre_matrix[-1,], stringsAsFactors=FALSE) #re
36 for (c in 1:ncol(genre_matrix2)) {
37   genre_matrix2[,c] <- as.integer(genre_matrix2[,c])
38 }
39
40 #Create a matrix to search for a movie by genre:
41 years <- as.data.frame(movies$title, stringsAsFactors=FALSE)
42 library(data.table)
43 substring <- function(x, n){
44   substr(x, nchar(x)-n+1, nchar(x))
45 }
```

Environment

- genre_matr_ Large matrix (185940 elemen...
- genres 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables

```
29   genmat_col = which(genre_matrix[1,] == genres2[i,c])
30   genre_matrix[i+1,genmat_col] <- 1
31 }
32 }
33
34 #convert into dataframe
35 genre_matrix2 <- as.data.frame(genre_matrix[-1,], stringsAsFactors=FALSE) #re
36 for (c in 1:ncol(genre_matrix2)) {
37   genre_matrix2[,c] <- as.integer(genre_matrix2[,c])
38 } #convert from characters to integers
39
40 #Create a matrix to search for a movie by genre:
41 years <- as.data.frame(movies$title, stringsAsFactors=FALSE)
42 library(data.table)
43 substring <- function(x, n){
44   substr(x, nchar(x)-n+1, nchar(x))
45 }
```

Environment

- genre_matr_ Large matrix (185940 elemen...
- genre_matr_ 10329 obs. of 18 variables
- genres 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables

	Action	Adventure	Animation	Children	Comedy	Crime	Documentary	Drama	Fantasy
1	0	1	1	1	1	0	0	0	1
2	0	1	0	1	0	0	0	0	1
3	0	0	0	0	1	0	0	0	0
4	0	0	0	0	1	0	0	1	0
5	0	0	0	0	1	0	0	0	0
6	1	0	0	0	0	1	0	0	0
7	0	0	0	0	1	0	0	0	0
8	0	1	0	1	0	0	0	0	0
9	1	0	0	0	0	0	0	0	0

Environment

- genre_matr_ Large matrix (185940 elemen...
- genre_matr_ 10329 obs. of 18 variables
- genres 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables

So, if I just run this thing quickly and then drop the first row, see first row I am dropping here, if I just drop the first row and what do I get in genre matrix2? I am getting genre matrix 2 like this. So, the first movie is action, animation, children, comedy, the second movie is adventure, children and fantasy and so on. Now, this part, you can do it in different way. If you have a better algo for that, you can do it in your way. This part is similar so, I have to create this, this is something that I want to know that which movie and with genre I am creating a mapping for that.

(Refer Slide Time: 7:16)

The screenshot shows the RStudio interface with the following code in the editor:

```

29 genmat_col = which(genre_matrix[1,] == genres2[f,c])
30 genre_matrix[1+1,genmat_col] <- 1
31
32 }
33
34 #convert into dataframe
35 genre_matrix2 <- as.data.frame(genre_matrix[-1,], stringsAsFactors=FALSE) #re
36 for (c in 1:ncol(genre_matrix2)) {
37   genre_matrix2[,c] <- as.integer(genre_matrix2[,c])
38 } #convert from characters to integers
39
40 #Create a matrix to search for a movie by genre:
41 years <- as.data.frame(movies$title, stringsAsFactors=FALSE)
42 library(data.table)
43 substrRight <- function(x, n){
44   substr(x, nchar(x)-n+1, nchar(x))
45 }

```

The Environment pane on the right shows the following objects:

- genre_matr_ 10329 obs. of 18 variables
- genr_1s 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables

The Console shows the execution of the code above.

The screenshot shows the RStudio interface with the following code in the editor:

```

33
34 #convert into dataframe
35 genre_matrix2 <- as.data.frame(genre_matrix[-1,], stringsAsFactors=FALSE) #re
36 for (c in 1:ncol(genre_matrix2)) {
37   genre_matrix2[,c] <- as.integer(genre_matrix2[,c])
38 } #convert from characters to integers
39
40 #Create a matrix to search for a movie by genre:
41 years <- as.data.frame(movies$title, stringsAsFactors=FALSE)
42 library(data.table)
43 substrRight <- function(x, n){
44   substr(x, nchar(x)-n+1, nchar(x))
45 }
46 years <- as.data.frame(substr(substrRight(substrRight(years$movies$title', 6
47
48 search_matrix <- cbind(movies[,1], substr(movies[,2],1,nchar(movies[,2])-6),
49

```

The Environment pane on the right shows the following objects:

- genre_matr_ Large matrix (185940 elemen
- genre_matr_ 10329 obs. of 18 variables
- genres 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables

The Console shows the execution of the code above.

Now, next is for c in 1 to 18 n column genre matrix2 is 18, so, c is 1 to 18 the genre matrix for each column I am changing it to their integer values. So, if I just run this, is doing nothing but changing all values to their integer values. Now, what will I do? I will check, create a search for a movie by genre so, this is an off topic, but this is something that helps me in developing further. So, I am creating a data frame called years which is the movie title. So, years is equal to as data frame movie titles, string as false so, years looks this, so, this is the movie title column basically.

And then I am using library called data table, to what to do? To substring so, I am saying that substring write is a function, which will substring from left to right. So, this is what I am doing substring function and using the substring function from here, I will only take the years this 1995, this 1995, this 1995 I will scrape so, you can do right how much? From right side it is like the right function, if you have used Excel, it is the right function, on the right it is the second to 1, 2, 3, 4, fifth. So, it should start from 2 in that 5. So, from right side that is what I am doing.

So years is equal to you see that 1 , 4 basically is something that it is doing, it is subtracting, creating a subset and that years is getting me all this year values of this particular thing. So again, you can do it in Excel if you want, but I am doing it here.

(Refer Slide Time: 9:18)

The screenshot shows the RStudio interface with the following R code in the script editor:

```
40 #create a matrix to search for a movie by genre:
41 years <- as.data.frame(movies$title, stringsAsFactors=FALSE)
42 library(data.table)
43 substrRight <- function(x, n){
44   substr(x, nchar(x)-n+1, nchar(x))
45 }
46 years <- as.data.frame(substr(substrRight(substrRight(years$'movies$title', 6), 5),
47 1,4))
48 search_matrix <- cbind(movies[,1], substr(movies[,2],1,nchar(movies[,2])-6),
49 colnames(search_matrix) <- c("movieid", "title", "year", "genre_list")
50
51 write.csv(search_matrix, "search.csv")
52 search_matrix <- read.csv("search.csv", stringsAsFactors=FALSE)
53
54 # Example of search an Action movie produced in 1995:
55 subset(search_matrix, Action == 1 & year == 1995)$title
56
```

The Environment pane on the right shows the following objects:

- genre_matr_ 10329 obs. of 18 variables
- genres 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables
- years 10329 obs. of 1 variable

The Console shows the execution of the following commands:

```
> view(years)
> library(data.table)
> substrRight <- function(x, n){
+   substr(x, nchar(x)-n+1, nchar(x))
+ }
> view(years)
> years <- as.data.frame(substr(substrRight(substrRight(years$'movies$title', 6), 5),
+ 1,4))
> view(years)
>
```

The screenshot shows the RStudio interface with a data table of movies. The table has the following columns: movieid, title, and genres.

movieid	title	genres
1	1. Toy Story (1995)	Adventure Animation Children Comedy Fantasy
2	2. Jumanji (1995)	Adventure Children Fantasy
3	3. Grumpier Old Men (1995)	Comedy Romance
4	4. Waiting to Exhale (1995)	Comedy Drama Romance
5	5. Father of the Bride Part II (1995)	Comedy
6	6. Heat (1995)	Action Crime Thriller
7	7. Sabrina (1995)	Comedy Romance
8	8. Tom and Huck (1995)	Adventure Children
9	9. Sudden Death (1995)	Action
10	10. GoldenEye (1995)	Action Adventure Thriller

The Environment pane on the right shows the same objects as in the previous screenshot.

The Console shows the execution of the following commands:

```
> library(data.table)
> substrRight <- function(x, n){
+   substr(x, nchar(x)-n+1, nchar(x))
+ }
> view(years)
> years <- as.data.frame(substr(substrRight(substrRight(years$'movies$title', 6), 5),
+ 1,4))
> view(years)
> view(movies)
>
```

The screenshot shows the RStudio interface with the following R code in the script editor:

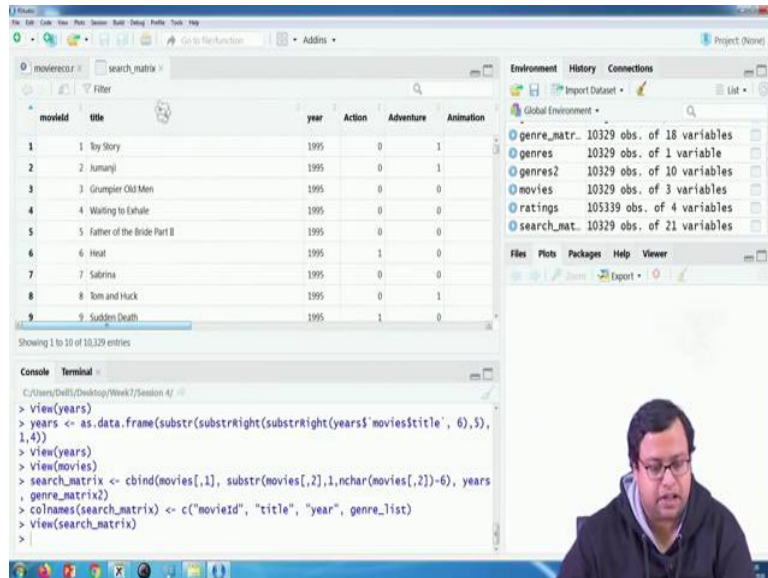
```
40 #create a matrix to search for a movie by genre:
41 years <- as.data.frame(movies$title, stringsAsFactors=FALSE)
42 library(data.table)
43 substrRight <- function(x, n){
44   substr(x, nchar(x)-n+1, nchar(x))
45 }
46 years <- as.data.frame(substr(substrRight(substrRight(years$'movies$title', 6),
47 1,4))
48 search_matrix <- cbind(movies[,1], substr(movies[,2],1,nchar(movies[,2])-6),
49 colnames(search_matrix) <- c("movieid", "title", "year", "genre_list")
50
51 write.csv(search_matrix, "search.csv")
52 search_matrix <- read.csv("search.csv", stringsAsFactors=FALSE)
53
54 # Example of search an Action movie produced in 1995:
55 subset(search_matrix, Action == 1 & year == 1995)$title
56
```

The Environment pane on the right shows the following objects:

- genre_matr_ 10329 obs. of 18 variables
- genres 10329 obs. of 1 variable
- genres2 10329 obs. of 10 variables
- movies 10329 obs. of 3 variables
- ratings 105339 obs. of 4 variables
- years 10329 obs. of 1 variable

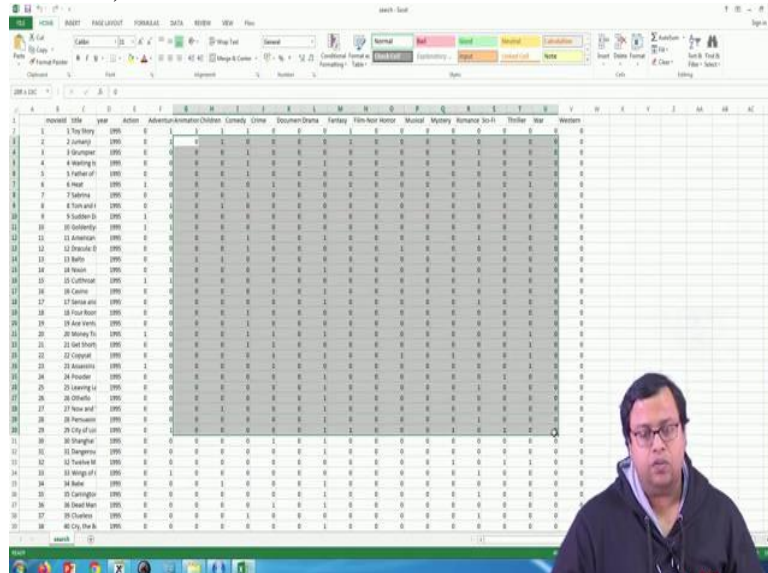
The Console shows the execution of the following commands:

```
> library(data.table)
> substrRight <- function(x, n){
+   substr(x, nchar(x)-n+1, nchar(x))
+ }
> view(years)
> years <- as.data.frame(substr(substrRight(substrRight(years$'movies$title', 6), 5),
+ 1,4))
> view(years)
> view(movies)
>
```



Next is I will create the same thing with a movie. So I am getting a search matrix, which is binding movies data first column. That means movie ID, next is substring of movies one, two, so the movie name, basically only the movie name, then the years and then the genre matrix. So, when I put this and then put the whole column names, this is the search matrix that I get a movie ID, the title, the year, and then the genre id matrix. So, this is what I have created from this. I save this so that I can use it for later purpose.

(Refer Slide Time: 10:06)



```

48 search_matrix <- cbind(movies[,1], substr(movies[,2],1,nchar(movies[,2])-6),
49 colnames(search_matrix) <- c("movieid", "title", "year", genre_list)
50
51 write.csv(search_matrix, "search.csv")
52 search_matrix <- read.csv("search.csv", stringsAsFactors=FALSE)
53
54 # Example of search an Action movie produced in 1995:
55 subset(search_matrix, Action == 1 & year == 1995)$title
56
57 ## Create a user profile
58 binaryratings <- ratings
59
60 # ratings of 4 and 5 are mapped to 1,
61 # representing likes, and ratings of 3
62 # and below are mapped to -1, representing
63 # dislikes:
64
556 (Dip Level)

```

Console Terminal

```

[38] "Operation Dumbo Drop "
[39] "Best of the Best 3: No Turning Back "
[40] "Tokyo Fist (Tokyo ken) "
[41] "Iron Eagle IV "
[42] "3 Ninjas Knuckle Up "
[43] "Crying Freeman "
[44] "Ninja Scroll (Jûbei ninpûchû) "
[45] "Darkman II: Return of Durant, the "
[46] "Blade, The (Ooo) "
>

```

```

48 search_matrix <- cbind(movies[,1], substr(movies[,2],1,nchar(movies[,2])-6),
49 colnames(search_matrix) <- c("movieid", "title", "year", genre_list)
50
51 write.csv(search_matrix, "search.csv")
52 search_matrix <- read.csv("search.csv", stringsAsFactors=FALSE)
53
54 # Example of search an Action movie produced in 1995:
55 subset(search_matrix, Action == 1 & year == 1995)$title
56
57 ## Create a user profile
58 binaryratings <- ratings
59
60 # ratings of 4 and 5 are mapped to 1,
61 # representing likes, and ratings of 3
62 # and below are mapped to -1, representing
63 # dislikes:
64
556 (Dip Level)

```

Console Terminal

```

> subset(search_matrix, Action == 1 & Comedy==1 & year == 1995)$title
[1] "Money Train "
[2] "Rumble in the Bronx (Hont faan kut) "
[3] "Bad Boys "
[4] "Batman Forever "
[5] "French Kiss "
[6] "Tank Girl "
[7] "Operation Dumbo Drop "
>

```

So this is how it looks like basically, the search matrix will look like movie ID, title, year and the overall genre of the movies. So, you can create a pivot in this particular matrix or you can create, you can search like this subset, search matrix, action is equal to 1 that means it is a movie action, and years = 1995, what is the title of that? If I find out, so these are the movies, which are, which come out in 1995 under action movies.

So if you want furthermore, let us say, I want action = 1 and comedy = 1 and 1995. So, these are the movies which are action and comedy. So, you can create those kind of search.

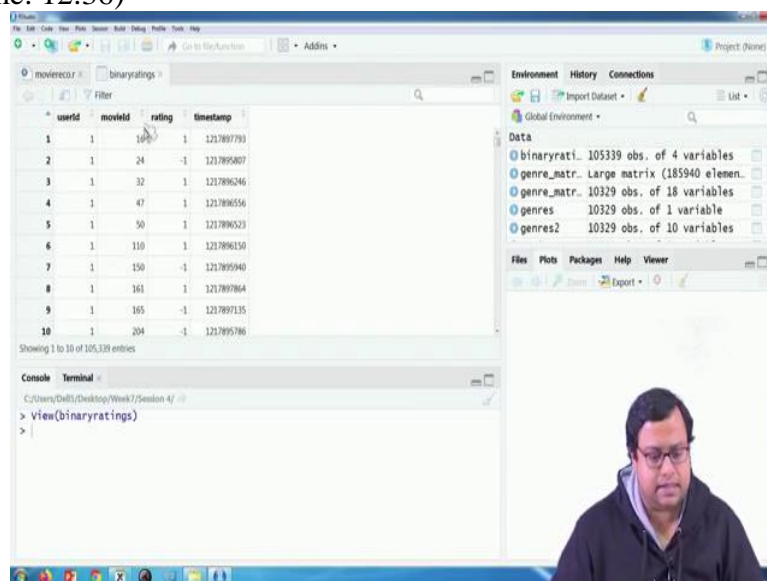
Now, what I will do next is I will create a binary rating. So, create a user profile is what is my next objective, so I am creating binary ratings anything 4 and 5 is high, 3, 2, 1 is low. So, that is what I am creating here so, binary rating is equal to ratings and then if the value is greater

than 3 then put it 1, if it is smaller than 3 then - 1. So, that is what I just trying to do. Likes means 1, dislikes means - 1 so, it will run for some time. And let us see how much time it runs.

So, i in 1 to n row binary ratings, binary ratings is 1 lakh. So, it will take quite a bit of time to totally find out so, I will see it is now 28,000. And what I am trying to do here, basically it will go up to 1 lakh. What I am going to do is that sometimes it does not matter if there are multiple gaps let us say 1, 2, 3, 4, 5, multiple levels of ratings, creating similarity becomes very difficult sometimes. So, 1 - 1 is basically a more or less I would say **dichotomy**. So, if you like 1, if you dislike - 1 and why did I take ≥ 4 , 4 or 5? Because oftentimes you have seen that the reviews are a little bit positively skewed in nowadays.

So, if it is positively skewed so, 1, 2, 3 is considered to be low and 4 and 5 is considered to be high and we are actually trying to do that we are in 98,000, 105000 will take okay. So, it is over so, I have created a binary rating. So, how does the binary ratings look like?

(Refer Slide Time: 12:36)



```

64
65- For (i in 1:nrow(binaryratings)){
66-   if (binaryratings[i,3] > 3){
67-     binaryratings[i,3] <- -1
68-   }
69-   else{
70-     binaryratings[i,3] <- -1
71-   }
72- }
73
74 # convert binaryratings matrix to the correct format:
75 binaryratings2 <- dcast(binaryratings, movieId~userId, value.var = "rating",
76
77- For (i in 1:ncol(binaryratings2)){
78-   binaryratings2[which(is.na(binaryratings2[,i]) == TRUE),i] <- 0
79- }
80
81 (Dip Level)

```

Environment

- Global Environment
- Data
 - binaryrati_105339 obs. of 4 variables
 - genre_matr_ Large matrix (185940 elemen
 - genre_matr_ 10329 obs. of 18 variables
 - genres 10329 obs. of 1 variable
 - genres2 10329 obs. of 10 variables

Console

```

C:\Users\Delhi\Desktop\Week7\Session 4\7 >
> View(binaryratings)
>

```

Environment

- Global Environment
- Data
 - binaryrati_105339 obs. of 4 variables
 - binaryrati_10325 obs. of 669 variables
 - genre_matr_ Large matrix (185940 elemen
 - genre_matr_ 10329 obs. of 18 variables
 - genres 10329 obs. of 1 variable

Console

```

C:\Users\Delhi\Desktop\Week7\Session 4\7 >
> binaryratings2 <- dcast(binaryratings, movieId~userId, value.var = "rating", na.rm
=FALSE)
Warning message:
In dcast(binaryratings, movieId ~ userId, value.var = "rating", :
The dcast generic in data.table has been passed a data.frame and will attempt to r
edirect to the reshape2::dcast; please note that reshape2 is deprecated, and this r
edirection is now deprecated as well. Please do this redirection yourself like reshap
e2:::dcast(binaryratings). In the next version, this warning will become an error.
> View(binaryratings2)
>

```

```

68 }
69- else{
70-   binaryratings[i,3] <- -1
71- }
72- }
73
74 # convert binaryratings matrix to the correct format:
75 binaryratings2 <- dcast(binaryratings, movieId~userId, value.var = "rating",
76
77- For (i in 1:ncol(binaryratings2)){
78-   binaryratings2[which(is.na(binaryratings2[,i]) == TRUE),i] <- 0
79- }
80 binaryratings2 = binaryratings2[,-1] #remove movieids col. rows are movieids,
81
82 #Remove rows that are not rated from movies dataset
83
84
85 (Dip Level)

```

Environment

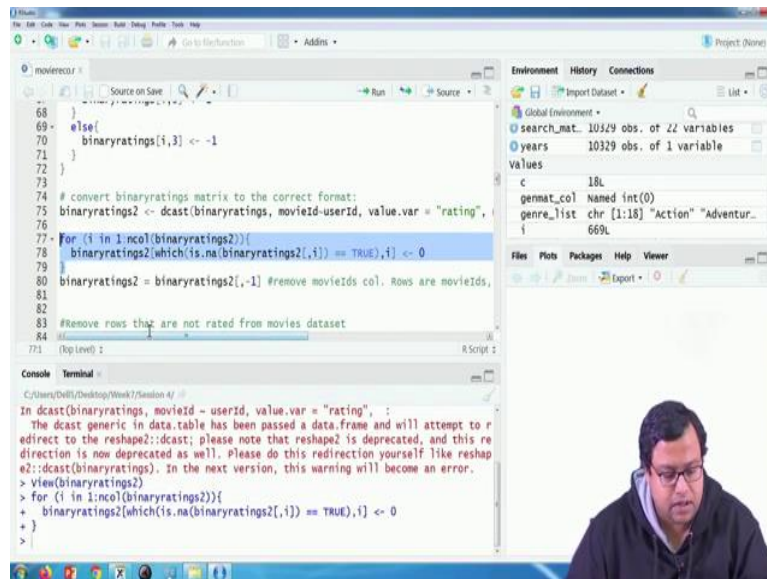
- Global Environment
- Data
 - binaryrati_105339 obs. of 4 variables
 - binaryrati_10325 obs. of 669 variables
 - genre_matr_ Large matrix (185940 elemen
 - genre_matr_ 10329 obs. of 18 variables
 - genres 10329 obs. of 1 variable

Console

```

C:\Users\Delhi\Desktop\Week7\Session 4\7 >
> # convert binaryratings matrix to the correct format:
> binaryratings2 <- dcast(binaryratings, movieId~userId, value.var = "rating", na.rm
=FALSE)
Warning message:
In dcast(binaryratings, movieId ~ userId, value.var = "rating", :
The dcast generic in data.table has been passed a data.frame and will attempt to r
edirect to the reshape2::dcast; please note that reshape2 is deprecated, and this r
edirection is now deprecated as well. Please do this redirection yourself like reshap
e2:::dcast(binaryratings). In the next version, this warning will become an error.
>

```



Binary ratings look like this, basic thing 1 or - 1 nothing else. So, this code is actually binary code you could have done a if else, simple if else would have taken much lesser time, anyways. So, then what I do is convert this binary rating so, what I will do is I will convert binary matrix with correct format.

So, the format is like this binary ratings2 which I run and it will create a binary ratings2, which is the movie ID. And then correspondingly, there are lots of people 669 variables means 669 observations, all these NS means nobody has given review for that. And if they have given some review either it is positive or negative, 1 means positives, - 1 is negative. So, that is what I have got so, basically 10,325 movies and 669 users, I will got a 0 1 matrix for that. So, it is a large matrix actually and then, so, if by chance, if there is nothing there, then if there is NA, then put a 0 there basically.

So, I am putting 0 for all the values where it is NA and then remove the movie ID column. So, that means I am getting up 10,325 observations, which is the movies, 668 observations, which are the users and 0 1 and - 1 this is the three values that are there, 0 means there was no rating, 1 means he liked it, - 1 means he basically disliked so, that is a binary rating formula that I have created.

(Refer Slide Time: 14:14)

```
75 binaryratings2 <- dcast(binaryratings, movieId~userId, value.var = "rating",
76
77 - for (i in 1:ncol(binaryratings2)){
78   binaryratings2[which(is.na(binaryratings2[,i]) == TRUE),i] <- 0
79 }
80 binaryratings2 = binaryratings2[,-1] #remove movieIds col. Rows are movieIds,
81
82
83 #Remove rows that are not rated from movies dataset
84 movieIds <- length(unique(movies$movieId)) #10329
85 ratingmovieIds <- length(unique(ratings$movieId)) #10325
86 movies2 <- movies[which(movies$movieId %in% ratings$movieId) == FALSE,]
87 rownames(movies2) <- NULL
88
89 #Remove rows that are not rated from genre_matrix2
90 genre_matrix3 <- genre_matrix2[which(movies$movieId %in% ratings$movieId) ==
91 FALSE,]
92 rownames(genre_matrix3) <- NULL
93
94 # calculate the dot product of the genre matrix and
95 # the ratings matrix and obtain the user profiles
```

Environment History Connections
Global Environment
Data
binaryrati_ 105339 obs. of 4 variables
binaryrati_ 10325 obs. of 668 variables
genre_matr_ Large matrix (185940 elemen
genre_matr_ 10329 obs. of 18 variables
genres 10329 obs. of 1 variable

Console Terminal
C:/Users/Dell/Desktop/Week7/Session 4/ /
eDirect to the reshape2::dcast; please note that reshape2 is deprecated, and this re-
direction is now deprecated as well. Please do this redirection yourself like reshap
e2::dcast(binaryratings). In the next version, this warning will become an error.
> view(binaryratings2)
> for (i in 1:ncol(binaryratings2)){
+ binaryratings2[which(is.na(binaryratings2[,i]) == TRUE),i] <- 0
+ }
> binaryratings2 = binaryratings2[,-1] #remove movieIds col. Rows are movieIds, cols
> are users

```
80 binaryratings2 = binaryratings2[,-1] #remove movieIds col. Rows are movieIds,
81
82
83 #Remove rows that are not rated from movies dataset
84 movieIds <- length(unique(movies$movieId)) #10329
85 ratingmovieIds <- length(unique(ratings$movieId)) #10325
86 movies2 <- movies[which(movies$movieId %in% ratings$movieId) == FALSE,]
87 rownames(movies2) <- NULL
88
89 #Remove rows that are not rated from genre_matrix2
90 genre_matrix3 <- genre_matrix2[which(movies$movieId %in% ratings$movieId) ==
91 FALSE,]
92 rownames(genre_matrix3) <- NULL
93
94 # calculate the dot product of the genre matrix and
95 # the ratings matrix and obtain the user profiles
```

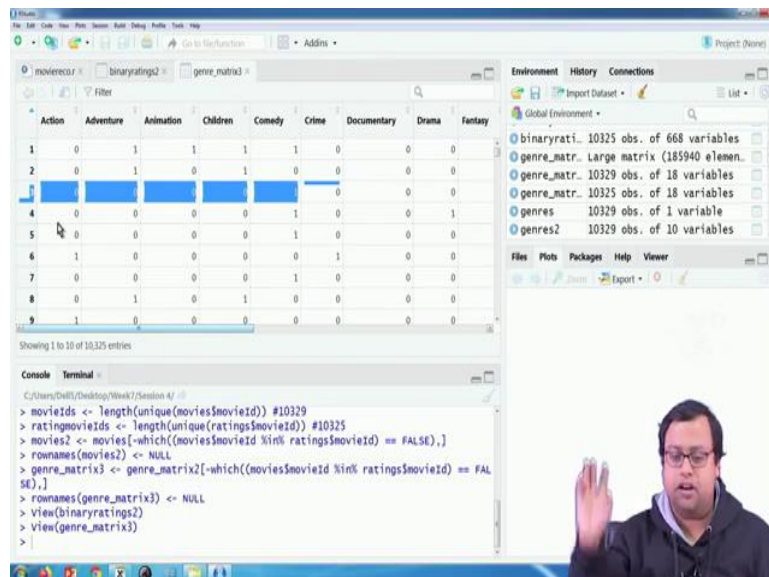
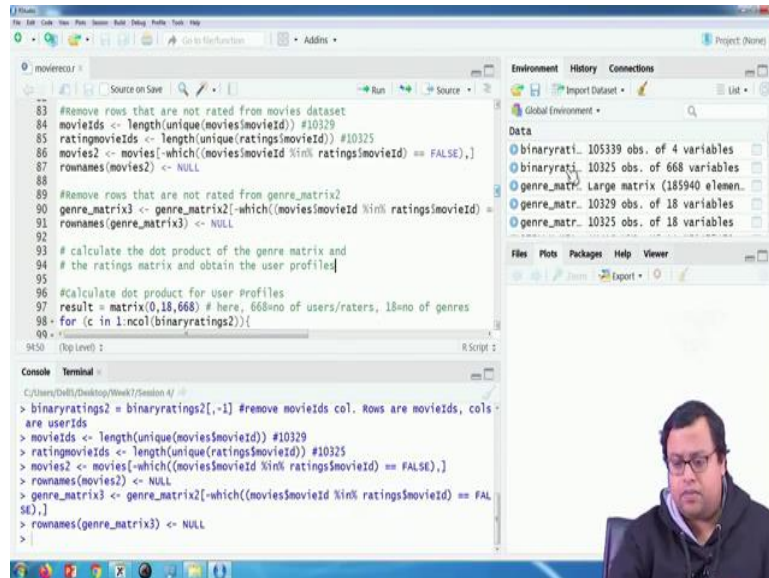
Environment History Connections
Global Environment
Data
binaryrati_ 105339 obs. of 4 variables
binaryrati_ 10325 obs. of 668 variables
genre_matr_ Large matrix (185940 elemen
genre_matr_ 10329 obs. of 18 variables
genres 10329 obs. of 1 variable

Console Terminal
C:/Users/Dell/Desktop/Week7/Session 4/ /
> binaryratings2 = binaryratings2[,-1] #remove movieIds col. Rows are movieIds, cols
are users
> movieIds <- length(unique(movies\$movieId)) #10329
> ratingmovieIds <- length(unique(ratings\$movieId)) #10325
> movies2 <- movies[which(movies\$movieId %in% ratings\$movieId) == FALSE,]
> rownames(movies2) <- NULL
> genre_matrix3 <- genre_matrix2[which(movies\$movieId %in% ratings\$movieId) == FAL
SE,]
> rownames(genre_matrix3) <- NULL
>

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
10	0	0	0	0	0	0	0	1	0	-1	0	-1	0	0	0	0	0
11	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	1	0	0	0	0	0	0	0	1	0	0	-1	0	0	0	0	0
17	0	1	0	0	0	0	0	0	0	0	0	-1	0	0	1	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Environment History Connections
Global Environment
Data
binaryrati_ 10325 obs. of 668 variables
genre_matr_ Large matrix (185940 elemen
genre_matr_ 10329 obs. of 18 variables
genres 10329 obs. of 1 variable
genres2 10329 obs. of 10 variables

Console Terminal
C:/Users/Dell/Desktop/Week7/Session 4/ /
are users
> movieIds <- length(unique(movies\$movieId)) #10329
> ratingmovieIds <- length(unique(ratings\$movieId)) #10325
> movies2 <- movies[which(movies\$movieId %in% ratings\$movieId) == FALSE,]
> rownames(movies2) <- NULL
> genre_matrix3 <- genre_matrix2[which(movies\$movieId %in% ratings\$movieId) == FAL
SE,]
> rownames(genre_matrix3) <- NULL
> view(binaryratings2)
>



Now remove rows that are not rated from movies datasets, there are certain movies which are absolutely not rated by anybody. So, if they are not rated by anybody, why should I use them? So, we are doing that and then movies rows that are not rated for genre matrix2 also so, any movies which has been not rated you have to remove that from genre matrix2 also. That is why I am creating genre matrix3, so, genre matrix2 to 3 there are four movies, which were not rated 10,329 10,225 so, that means there are four movies which were not rated, we removed them.

Similarly movies2 basically have 10,325, therefore movies which were not rated by anybody, we removed them. Fair enough, that is how we are reducing the dataset, so the calculation takes less time. Then what, calculate the dot product of the genre matrix and the ratings

matrix and obtain the user profile. So, understand what is the genre matrix. Genre matrix is 10,325 of 18 genres, whether this movie is belonging to this genre, and then what is my rating matrix? Rating matrix is basically 10,325 of various people giving a rating.

Now, if I want to find out that whether this user is fond of comedy, then what will I do? I have seen that this user, this 668 from this binary matrix, this one, if I just click on this one, let us see user number one. Let it come and then I will say, let us say user number one whether he so, user number one if you say that 10,325 or let us says user number two because the data is showing here, he has seen this movie, he has not liked this movie, he has not liked this movie and if you come down further he has liked this movie. So, some movies user number 2 like, some movies user number 2 did not like.

And then all these movies, like movie number 14, if I go to genre matrix, I know that movie number 14 is basically a, movie number 14, movie number 14 is basically is a drama movie, which he liked. And then it is only a drama probably.

Then, let us say this one, which he did not like, which one was that? That was movie number 3, or movie number 5 he did not like. So, in the genre matrix I know movie number 3 is basically a comedy, movie number 5 is also comedy. And movie number 3 is nothing else. Probably romance also, movie number 5 is not romance. So, movie number 3 and 5 both are comedy which the user number 2 did not like, so from this information I can find out what is that he is not liking, the overall liking of a genre, how would I do that? Which movies he has seen, which genre it is from there, I will find out how much is rated for them, + 1 or - 1.

(Refer Slide Time: 17:34)

```
91 rownames(genre_matrix3) <- NULL
92
93 # calculate the dot product of the genre matrix and
94 # the ratings matrix and obtain the user profiles
95
96 #calculate dot product for user profiles
97 result = matrix(0,18,668) # here, 668=no of users/raters, 18=no of genres
98 for (c in 1:ncol(binaryratings2)){
99   for (i in 1:ncol(genre_matrix3)){
100     result[i,c] <- sum((genre_matrix3[,i]) * (binaryratings2[,c])) #ratings p
101   }
102 }
103
104 #convert to binary scale
105 for (c in 1:ncol(result)){
106   for (i in 1:nrow(result)){
107     result[i,c] <- ifelse(result[i,c] > 0, 1, -1)
108   }
109 }
110 }
```

Environment: movies2 (10325 obs. of 3 variables), ratings (105339 obs. of 4 variables), result (num [1:18, 1:668] 28 15 2 1), search_mat_ (10329 obs. of 22 variables), years (10329 obs. of 1 variable)

```
> view(binaryratings2)
> view(genre_matrix3)
> result = matrix(0,18,668)
> for (c in 1:ncol(binaryratings2)){
+   for (i in 1:ncol(genre_matrix3)){
+     result[i,c] <- sum((genre_matrix3[,i]) * (binaryratings2[,c])) #ratings per ge
+   }
+ }
>
```

```
> view(result)
```

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18			
1	28	7	-7	6	1	6	37	1	-29	0	3	-2	0	-2	13	30	43	-1	1	9	29
2	15	8	-1	11	12	12	5	2	-15	0	8	-2	1	1	16	11	48	1	-10	7	41
3	2	2	0	4	19	4	2	1	-1	2	3	1	-2	1	3	0	10	4	-2	1	15
4	1	3	1	4	17	6	-3	-2	-4	2	-3	1	0	3	3	-3	8	2	-2	-1	3
5	1	1	1	30	19	18	14	0	-41	2	-10	2	4	-2	26	-3	51	6	8	4	11
6	27	1	2	12	-2	7	19	0	-11	2	-4	-1	3	1	12	6	12	10	8	6	3
7	-1	0	1	0	0	0	1	0	-1	0	0	0	0	0	0	0	0	-1	0	2	0
8	27	9	12	66	1	23	9	16	-33	7	-8	-2	7	-2	37	9	25	6	54	11	22
9	4	4	-1	6	12	2	-3	0	-6	0	1	4	0	1	15	2	16	7	-9	4	28
10	2	0	0	6	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0

```
>
```

So that is what I am going to do here in the next set of calculations. So, I am creating a result which is 18 , 668, why 18? That means 18 rows, 668 columns, each column is one consumer or one user. And I am saying that whether he, basically a sum of 1's and - 1's, whether he has seen a movie in a particular genre, if like liked 1, if he disliked - 1. If the net sum is positive, then overall he is liking, if the net sum is negative then overall he disliked in that genre. So, that is what I am populating here, quickly so, result, I got this result column.

So now, if I just see this matrix carefully, what I get is user number 2 do not, user number 1 likes the first genre most 28, then sixth genre, the eighth genre and probably 16th genre. These are the genres that user number 1 likes, user number 2, probably this one, second genre, ninth genre. Now, if by chance see there are two guys who are, it is user number 1 is a

movie buff, he watches more movie, so the one which one he dislikes, likes a ninth is still much higher than the most liked one for the second user, which is 9 here. So he is 9 and this 9 has two different meaning. For the user 2 this 9 means that he likes the eighth genre most but he does not watch movie much.

On the other hand, this 9 means that he watches movie very much. 9 rating is actually in comparison to 28, 27, 27 they are very small so, it is not his favorite genre, though the overall rating is 9. So, how to check that situation I have to normalize that. So, I am doing that, first of all convert the, so, I am converting them to 0 and 1, if it is smaller than 0, 0 if it is greater than 0, 1. Now I am creating a dataset, which will be used by this recommended engine, that recommender lab library to create the recommendation engine.

(Refer Slide Time: 19:49)

The screenshot shows the RStudio interface with the following R code in the script editor:

```
115
116 ## Assume that users like similar items, and retrieve movies
117 ## that are closest in similarity to a user's profile, which
118 ## represents a user's preference for an item's feature.
119 ## use Jaccard Distance to measure the similarity between user profiles
120
121 # The User-based collaborative Filtering Approach
122
123 library(reshape2)
124 #Create ratings matrix. Rows = userId, columns = movieId
125 ratingmat <- dcast(ratings, userId~movieId, value.var = "rating", na.rm=FALSE)
126 ratingmat <- as.matrix(ratingmat[, -1]) #remove userIds
127
128 # Method: UBCF
129 # Similarity Calculation Method: Cosine Similarity
130 # Nearest Neighbors: 30
131
132 (Rp Level) :
```

The console output shows the following messages:

```
Attaching package: 'reshape2'

The following objects are masked from 'package:data.table':

  dcast, melt

> #Create ratings matrix. Rows = userId, Columns = movieId
> ratingmat <- dcast(ratings, userId~movieId, value.var = "rating", na.rm=FALSE)
> ratingmat <- as.matrix(ratingmat[, -1]) #remove userIds
>
```

The Environment pane on the right shows the following objects:

- Global Environment
- ratingmat: Large matrix (6897100 elements)
- ratings: 105339 obs. of 4 variables
- result: num [1:18, 1:668] 1 1 1 1 1
- search_mat: 10329 obs. of 22 variables
- years: 10329 obs. of 1 variable

The Values pane shows the following values:

```
###
###
```


The following objects are masked from 'package:data.table':

```

dcast, melt

> #create ratings matrix. Rows = userID, Columns = movieId
> ratingsmat <- dcast(ratings, userID~movieId, value.var = "rating", na.rm=FALSE)
> ratingsmat <- as.matrix(ratingsmat[,-1]) #remove userIDs
> view(ratingsmat)
>

```

Environment: ratingmat (Large matrix (6897100 elems)), ratings (105339 obs. of 4 variables), result (num [1:18, 1:668] 1 1 1 1), search_mat (10329 obs. of 22 variables), years (10329 obs. of 1 variable).

```

123 library(reshape2)
124 #create ratings matrix. Rows = userID, Columns = movieId
125 ratingsmat <- dcast(ratings, userID~movieId, value.var = "rating", na.rm=FALSE)
126 ratingsmat <- as.matrix(ratingsmat[,-1]) #remove userIDs
127
128 # Method: UBCF
129 # similarity Calculation Method: cosine Similarity
130 # Nearest Neighbors: 30
131
132 library(recommenderlab)
133 #convert rating matrix into a recommenderlab sparse matrix
134 ratingsmat <- as(ratingsmat, "realRatingMatrix")
135
136 # Determine how similar the first four users are with each other
137 # create similarity matrix
138 similarity_users <- similarity(ratingsmat[1:4, ],
139                               method = "cosine",
140                               which = "users")
141
142 as.matrix(similarity_users)
143
144 # compute similarity between
145 # the first four movies
146 similarity_items <- similarity(ratingsmat[, 1:4], method =
147
148 (Dip Level)

```

The following objects are masked from 'package:data.table':

```

dcast, melt

> #create ratings matrix. Rows = userID, Columns = movieId
> ratingsmat <- dcast(ratings, userID~movieId, value.var = "rating", na.rm=FALSE)
> ratingsmat <- as.matrix(ratingsmat[,-1]) #remove userIDs
> view(ratingsmat)
> library(recommenderlab)
>

```

Environment: ratingmat (Large matrix (6897100 elems)), ratings (105339 obs. of 4 variables), result (num [1:18, 1:668] 1 1 1 1), search_mat (10329 obs. of 22 variables), years (10329 obs. of 1 variable).

```

131 library(recommenderlab)
132 #convert rating matrix into a recommenderlab sparse matrix
133 ratingsmat <- as(ratingsmat, "realRatingMatrix")
134
135
136 # Determine how similar the first four users are with each other
137 # create similarity matrix
138 similarity_users <- similarity(ratingsmat[1:4, ],
139                               method = "cosine",
140                               which = "users")
141
142 as.matrix(similarity_users)
143
144 # compute similarity between
145 # the first four movies
146 similarity_items <- similarity(ratingsmat[, 1:4], method =
147
148 (Dip Level)

```

The following objects are masked from 'package:data.table':

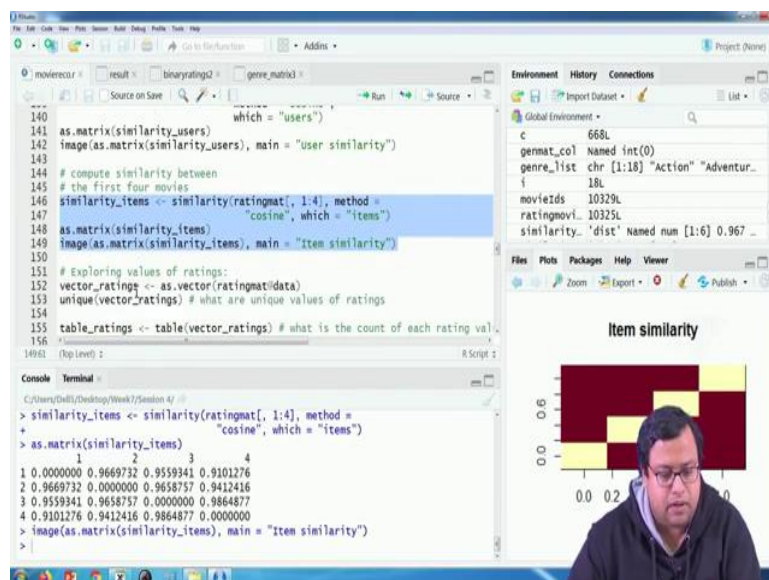
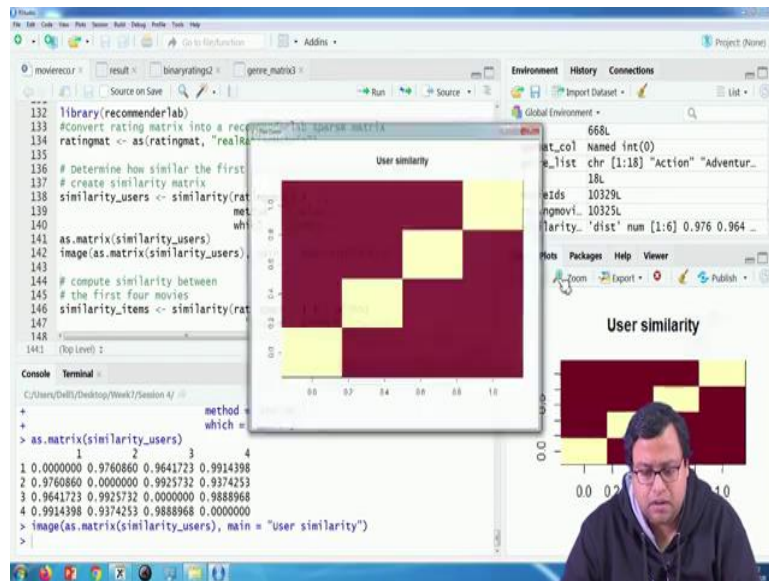
```

dcast, melt

> ratingsmat <- dcast(ratings, userID~movieId, value.var = "rating", na.rm=FALSE)
> ratingsmat <- as.matrix(ratingsmat[,-1]) #remove userIDs
> view(ratingsmat)
> library(recommenderlab)
> #convert rating matrix into a recommenderlab sparse matrix
> ratingsmat <- as(ratingsmat, "realRatingMatrix")
> similarity_users <- similarity(ratingsmat[1:4, ],
+                               method = "cosine",
+                               which = "users")
+

```

Environment: c (668L), genmat_col (Named int(0)), genre_list (chr [1:18] "Action" "Adventur...), I8L, movieIDs (10329L), ratingsmovi_ (10325L), similarity_ "dist" (num [1:6] 0.976 0.964 ...).



So, what I will do is I will create a rating mat, the library called reshape. And rating mat is getting created, rating matrix is a large matrix which looks like this. Let me just put it up here which is nothing but a reshaped version of the dataset that we have created till now. Let it come. So, if you check it carefully it is saying user ID to movie ID and the value is rating, user ID, movie ID and the value is rating, that is what is getting plotted here and the rating matrix all these 1's are basically the user IDs in x axis and y axis is movie id. So, it is taking a little bit higher time. So, the view is not coming properly, I will not focus on that.

And then the method is UBCF, see similarity calculation whether on cosine similarity that is what we are doing, nearest neighbors, remember, there we took five neighbors, here we are taking up to 30 neighbors, the library is recommender lab. The rating matrix is we are converting it to recommender labs parse matrix.

So, these are basically specific to this particular library you have to use this, you have no choice and then the similarity, basically similarity of the users is similarity rating mat 1 to 4, method is cosine which is user so, I am doing a user to users similarity matrix and if I try to find out a similarity score for the first four guys, this is how it looks like. So, you have taken first four, you can take the whole 1 to 668 it will give 1 to 668, I have taken the first four for the simplicity of calculation and this is what I am getting the image, the similarity matrix.

So, the yellowish it is the better, that is the image that we are getting so, this one is coming as to be good we have check the red colors again once more.

Then I can also find out the, compute similarity between the first four movies so, I can do it for the movie by movie similarity also so, for the first four movie if I try to find out the similarity, this is the similarity, see 0.9, 0.95, 0.91 so, the first four movies are very similar probably with each other.

Now, explore the value for ratings so, this is where I am actually trying to find out the ratings, the value of ratings so, vector ratings is equal to as vector, rating mat \$ data sorry, at the rate data so, this is a different format for sparse matrix and unique of vector rating. These are the various vector ratings that you got 5, 4, 3, 4.5, 1.2 and so on. And I am creating a table for that and it will just show me what is the tabular ratings so, vector ratings is these are the how many vector ratings that you have got is something that has been listed here.

(Refer Slide Time: 22:56)

The R console shows the following code:

```
table_ratings
# Visualize the rating:
vector_ratings <- vector_ratings[vector_ratings != 0] # rating == 0 are NA va
vector_ratings <- factor(vector_ratings)
qplot(vector_ratings) +
  ggtitle("Distribution of the ratings")
# Exploring viewings of movies:
views_per_movie <- colCounts(ratingmat) # count views for each movie
table_views <- data.frame(movie = names(views_per_movie),
  views = views_per_movie) # create dataframe of view
table_views <- table_views[order(table_views$views,
  decreasing = TRUE), ] # sort by number of vi
```

The Environment pane shows the following objects:

```
Global Environment
1RL
movieids 10329L
ratingmovi_ 10325L
similarity_ 'dist' Named num [1:6] 0.967
similarity_ 'dist' num [1:6] 0.976 0.964
table_rati_ 'table' int [1:11(id)] 679176
vector_rati_ Factor w/ 10 levels "0.5","1"
```

The plot shows a histogram titled "Distribution of the ratings" with the x-axis representing rating values (0.5, 1, 2, 2.5, 3, 3.5, 4, 4.5) and the y-axis representing the number of movies (0 to 30000).

The R console shows the following code:

```
views_per_movie <- colCounts(ratingmat) # count views for each movie
table_views <- data.frame(movie = names(views_per_movie),
  views = views_per_movie) # create dataframe of view
table_views <- table_views[order(table_views$views,
  decreasing = TRUE), ] # sort by number of vi
ggplot(table_views[1:6, ], aes(x = movie, y = views)) +
  geom_bar(stat="identity") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_x_discrete(labels=subset(movies2, movies2$movieId == table_views$movieI
  ggtitle("Number of views of the top movies")
#visualizing the matrix:
```

The Environment pane shows the following objects:

```
Global Environment
table_views 10325 obs. of 2 variables
years 10329 obs. of 1 variable
values
c 668L
genmat_col Named int(0)
genre_list chr [1:18] "Action" "Adventur_
1RL
```

The plot shows a bar chart titled "Number of views of the top movies" with the x-axis representing movie titles (Hulk: The Final Fight, Atomic Cafe, The Hitman, Marco Polo One Hundred Eyes (2016), NA, NA, NA) and the y-axis representing the number of views (0 to 300).

The R console shows the following code:

```
views_per_movie <- colCounts(ratingmat) # count views for each movie
table_views <- data.frame(movie = names(views_per_movie),
  views = views_per_movie) # create dataframe of view
table_views <- table_views[order(table_views$views,
  decreasing = TRUE), ] # sort by number of vi
ggplot(table_views[1:6, ], aes(x = movie, y = views)) +
  geom_bar(stat="identity") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_x_discrete(labels=subset(movies2, movies2$movieId == table_views$movieI
  ggtitle("Number of views of the top movies")
#visualizing the matrix:
```

The Environment pane shows the following objects:

```
Global Environment
table_views 10325 obs. of 2 variables
years 10329 obs. of 1 variable
values
c 668L
genmat_col Named int(0)
genre_list chr [1:18] "Action" "Adventur_
1RL
```

The plot shows a zoomed-in view of the bar chart titled "Number of views of the top movies" with the x-axis representing movie titles (Hulk: The Final Fight, Atomic Cafe, The Hitman, Marco Polo One Hundred Eyes (2016), NA, NA, NA) and the y-axis representing the number of views (0 to 300).

(Refer Slide Time: 23:54)

The RStudio interface shows the environment pane with variables: `ratingmat` (Large realRatingMatrix (1.9...)), `ratings` (105339 obs. of 4 variables), `result` (num [1:18, 1:668] 1 1 1 1), `search_mat` (10329 obs. of 22 variables), `table_views` (10325 obs. of 2 variables), and `years` (10329 obs. of 1 variable). The console shows the following code:

```
> ggplot(table_views[1:6, ], aes(x = movie, y = views)) +  
+ geom_bar(stat="identity") +  
+ theme(axis.text.x = element_text(angle = 45, hjust = 1)) +  
+ scale_x_discrete(labels=subset(movies2, movies2$movieid == table_views$movie)$title) +  
+ ggtitle("number of views of the top movies")  
> image(ratingmat[1:10, 1:15], main = "Heatmap of the first rows and columns")  
> view(ratingmat)
```

The heatmap, titled "Heatmap of the first rows and columns", shows a 10x15 grid of dark squares on a light background, representing the first 10 users and 15 items. The y-axis is labeled "Users (Rows)" and the x-axis is labeled "Items (Columns)". The dimensions are noted as "Dimensions: 10 x 15".

The RStudio interface shows the environment pane with variables: `ratingmat` (Large realRatingMatrix (1.9...)), `ratingmat_norm` (Large realRatingMatrix (1.9...)), `recommender_model` (Recommender(ratingmat_norm, method = "UBCF")), `search_mat` (10329 obs. of 22 variables), `table_views` (10325 obs. of 2 variables), and `years` (10329 obs. of 1 variable). The console shows the following code:

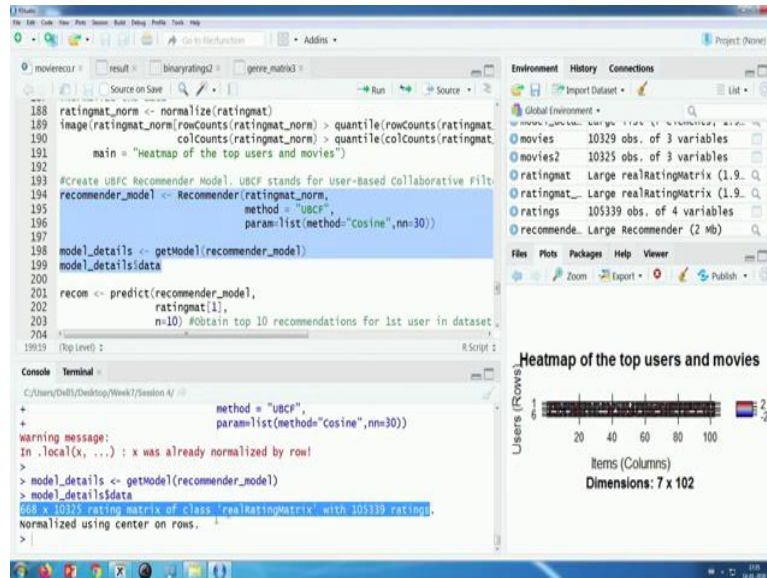
```
180 image(ratingmat, main = "Heatmap of the rating matrix") # hard to read- too ma  
181 image(ratingmat[1:10, 1:15], main = "Heatmap of the first rows and columns")  
182 image(ratingmat[rowCounts(ratingmat) > quantile(rowCounts(ratingmat), 0.99),  
183 colCounts(ratingmat) > quantile(colCounts(ratingmat), 0.99)],  
184 main = "Heatmap of the top users and movies")  
185  
186  
187 #Normalize the data  
188 ratingmat_norm <- normalize(ratingmat)  
189 image(ratingmat_norm[rowCounts(ratingmat_norm) > quantile(rowCounts(ratingmat_norm  
190 colCounts(ratingmat_norm) > quantile(colCounts(ratingmat_norm  
191 main = "Heatmap of the top users and movies")  
192  
193 #Create UBCF Recommender Model, UBCF stands for User-based collaborative Filtr  
194 recommender_model <- Recommender(ratingmat_norm,  
195 method = "UBCF",  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```

The heatmap, titled "Heatmap of the top users and movies", shows a 7x102 grid of dark squares on a light background, representing the top 7 users and 102 items. The y-axis is labeled "Users (Rows)" and the x-axis is labeled "Items (Columns)". The dimensions are noted as "Dimensions: 7 x 102".

The RStudio interface shows the environment pane with variables: `ratingmat` (Large realRatingMatrix (1.9...)), `ratingmat_norm` (Large realRatingMatrix (1.9...)), `recommender_model` (Recommender(ratingmat_norm, method = "UBCF")), `search_mat` (10329 obs. of 22 variables), `table_views` (10325 obs. of 2 variables), and `years` (10329 obs. of 1 variable). The console shows the following code:

```
180 image(ratingmat, main = "Heatmap of the rating matrix") # hard to read- too ma  
181 image(ratingmat[1:10, 1:15], main = "Heatmap of the first rows and columns")  
182 image(ratingmat[rowCounts(ratingmat) > quantile(rowCounts(ratingmat), 0.99),  
183 colCounts(ratingmat) > quantile(colCounts(ratingmat), 0.99)],  
184 main = "Heatmap of the top users and movies")  
185  
186  
187 #Normalize the data  
188 ratingmat_norm <- normalize(ratingmat)  
189 image(ratingmat_norm[rowCounts(ratingmat_norm) > quantile(rowCounts(ratingmat_norm  
190 colCounts(ratingmat_norm) > quantile(colCounts(ratingmat_norm  
191 main = "Heatmap of the top users and movies")  
192  
193 #Create UBCF Recommender Model, UBCF stands for User-based collaborative Filtr  
194 recommender_model <- Recommender(ratingmat_norm,  
195 method = "UBCF",  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```

The heatmap, titled "Heatmap of the top users and movies", shows a 7x102 grid of dark squares on a light background, representing the top 7 users and 102 items. The y-axis is labeled "Users (Rows)" and the x-axis is labeled "Items (Columns)". The dimensions are noted as "Dimensions: 7 x 102".



This is actually the one that I will use in my UBCF the recommender lab library. So, I am normalizing it first and after normalization this is the values that you are getting, the item columns and the user rows. And all these 1's are basically the heat map, what is a similarity level 2 to -2, various items and various rows how close it is, how is the chance of seeing this particular thing.

And then I use UBCF, UBCF means? User based collaborative filtering, and nearest number = 30, cosine is equal to, method is equal to cosine, you can change this cosine to correlation and etcetera. So, that is how you get the model and you find out the details of the data. So, the model details of the data says that there are 668 and 10,035 rating matrix of class, real rating matrix. With this many ratings has been used and this have normalized using center of rows.

(Refer Slide Time: 25:00)

The screenshot shows an R script in RStudio. The code defines a UBCF recommender model, predicts top 10 recommendations for the first user, and generates a heatmap. The console output shows the creation of a 668 x 10325 rating matrix and the resulting top 10 recommendations for the first user.

```
192 # Create UBCF Recommender Model, UBCF stands for user-based Collaborative Filtering
193 recommender_model <- Recommender(ratingmat_norm,
194   method = "UBCF",
195   param=list(method="cosine",nn=30))
196
197
198 model_details <- getModel(recommender_model)
199 model_details$data
200
201 recon <- predict(recommender_model,
202   ratingmat[1,],
203   n=10) #obtain top 10 recommendations for 1st user in dataset
204
205 recon
206
207 #recc_matrix <- sapply(recon$items,
208   function(x){ colnames(ratingmat)[x] })
209 #dim(recc_matrix)
210
211 recon_list <- as(recon,
212   "list") #convert recommenderlab object to readable list
213
214 #Obtain recommendations
215 recon_result <- matrix(0,10)
216 for (i in 1:10){
217   recon_result[i] <- as.character(subset(movies,
218     movies$movieid == as.integer(recon_list[[i]])))
219 }
220
221 # Evaluation:
222 evaluation_scheme <- evaluationScheme(ratingmat,
223   method="RMSE",
224   verbose=TRUE)
```

Environment: movies (10329 obs. of 3 variables), movies2 (10325 obs. of 3 variables), ratingmat (Large realRatingMatrix (1.9...)), ratingmat_ (Large realRatingMatrix (1.9...)), ratings (105339 obs. of 4 variables), recon (Large topNList (646.7 kb)).

Heatmap of the top users and movies. Dimensions: 7 x 102.

The screenshot shows the R script continuing with the conversion of the recommenderlab object to a readable list. The console output shows the same rating matrix and the top 10 recommendations for the first user.

```
211 recon_list <- as(recon,
212   "list") #convert recommenderlab object to readable list
213
214 #Obtain recommendations
215 recon_result <- matrix(0,10)
216 for (i in 1:10){
217   recon_result[i] <- as.character(subset(movies,
218     movies$movieid == as.integer(recon_list[[i]])))
219 }
220
221 # Evaluation:
222 evaluation_scheme <- evaluationScheme(ratingmat,
223   method="RMSE",
224   verbose=TRUE)
```

Environment: ratingmat_ (Large realRatingMatrix (1.9...)), ratings (105339 obs. of 4 variables), recon (Large topNList (646.7 kb)), reconme_ (Large Recommender (2 Mb)), result (num [1:18, 1:668] 1 1 1 1), search_mat_ (10329 obs. of 22 variables), table_views (10325 obs. of 2 variables).

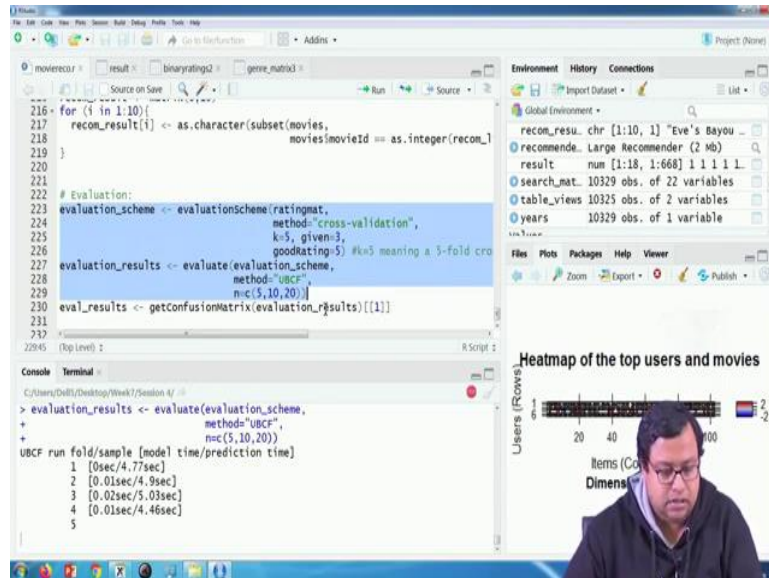
Heatmap of the top users and movies. Dimensions: 7 x 102.

The screenshot shows the final output of the R script, which is a list of the top 10 movie recommendations for the first user. The console output lists the movie titles.

```
[2.] "A.I. Artificial Intelligence (2001)"
[3.] "Sweet Hereafter, the (1997)"
[4.] "Being John Malkovich (1999)"
[5.] "Inception (2010)"
[6.] "One Flew Over the Cuckoo's Nest (1975)"
[7.] "Leaving Las Vegas (1995)"
[8.] "Thinner (1996)"
[9.] "It's Not Easy Bein' the (1997)"
[10.] "Gone Fishin' (1997)"
```

Environment: recon_resu_ (chr [1:10, 1] "Eve's Bayou..."), reconme_ (Large Recommender (2 Mb)), result (num [1:18, 1:668] 1 1 1 1), search_mat_ (10329 obs. of 22 variables), table_views (10325 obs. of 2 variables), years (10329 obs. of 1 variable).

Heatmap of the top users and movies. Dimensions: 7 x 102.



Now, if you want to find out a recommendation, the top 10 recommendation. So, for the first person I am taking rating mat 1, if you change this to 2 or 1 to 10, you will get the recommendations for all the 10 guys. So, recommendations for 10, it is the top 10 list for one user is something that is saved here in recom. So, if you check the recom, basically, the items and the ratings and the item levels have been written here.

So, the next part is basically if you want to see the recom, this is the recom list. So recom underscore list is basically the first person, the id of the various movies. And obtain the recommendations based on this recom list if I just tried to find out the names, then recom result will give me the names of the movies for the first guy. So, that is how based on UBCF you can check the code properly, we are doing it.

Now, if you have to evaluate and if I evaluate with $n = 1$, $n = 3$, see 1 nearest neighbor, 3 nearest neighbor, let us say 5, these are the three things that we are checking let us say 3, 5, 10 and 20. 5, 10, 20 these are the three guys I will check then how will I run it? I will just run it like this. So, good ratings = 5, bad rating so run this. First rating is taking 5 nearest neighbor, then it will take 10 nearest neighbor and then it will take 20 nearest neighbor and what is the evaluation score and etcetera is something that we will see right now.

(Refer Slide Time: 26:58)

The screenshot displays the RStudio environment. The main editor window contains R code for evaluating a recommendation engine. The code includes a function call to `as.character` to convert movie IDs to character format, followed by an `evaluation_scheme` object creation using `evaluationScheme` with parameters for cross-validation, k=5, given=3, and goodKating=5. The `evaluation_results` are generated using `evaluate`, and a confusion matrix is extracted with `getconfusionMatrix`. A reference link is provided: <https://rpubs.com/jeknov/movieRec>.

The console window shows the output of the `eval_results` command, displaying a table of performance metrics for different values of k (5, 10, 20):

	TP	FP	FN	TN	precision	recall	TPR
5	0.6617647	3.823529	22.97794	10294.54	0.1475410	0.04450541	0.04450541
10	1.0367647	7.933824	22.60294	10290.43	0.1155738	0.07316377	0.07316377
20	1.8823529	16.058824	21.75735	10282.30	0.1049180	0.13380915	0.13380915

Below the console, a heatmap titled "Heatmap of the top users and movies" is visible, showing the relationship between users and items. The x-axis is labeled "Items (Columns)" and the y-axis is labeled "Users (Rows)".

So, if I check that evaluation results, you will get it here so, this is the evaluation result for 5, 10 and 20 corresponding true positive, true negative rates and etcetera are giving here. So, the reference is this particular link, you can get a better discussion about the about this particular thing in this link, I have taken the code from that link and probably the dataset has been first publicly available. And this is how we create the recommendation engine with rating dataset with bigger dataset. In the next video, I will also show you how to create a recommendation engine with a smaller dataset which can still be handle able. Thank you very much. I will see you in the next video.