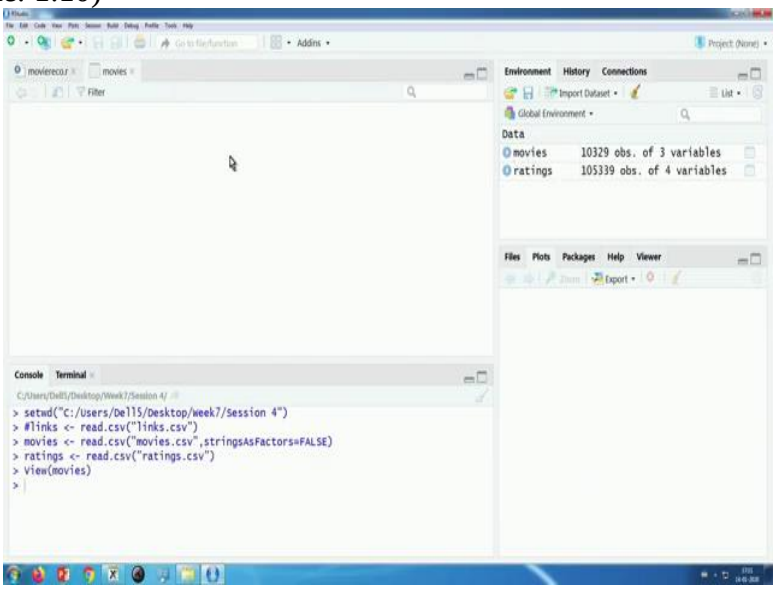
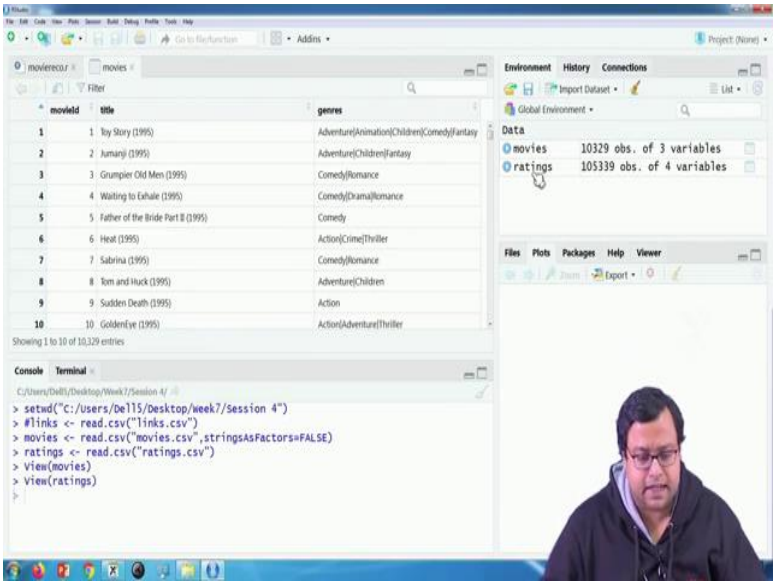
Marketing Analytics
Prof. Swagato Chatterjee
Vinod Gupta School of Management
Indian Institute of Technology, Kharagpur
Lecture 38: Recommendation Engine and Retail Analytics (Contd.)

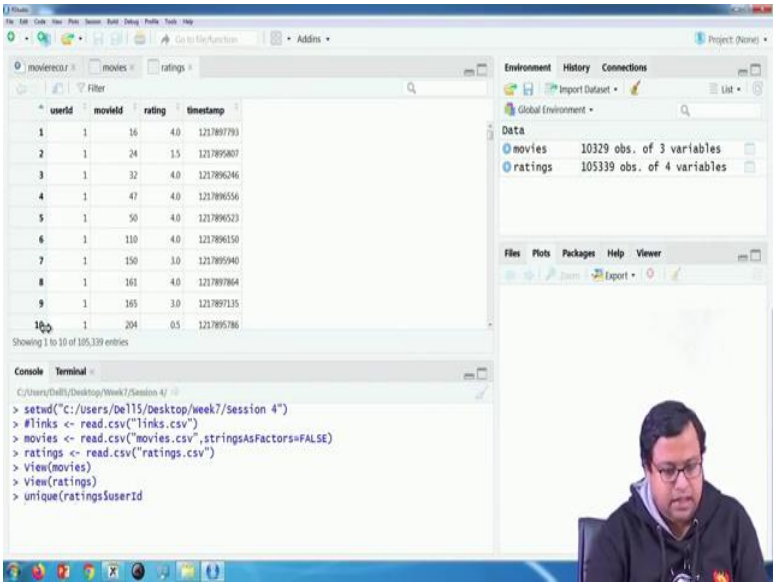
Hello, everybody, welcome to Marketing Analytics Course, this is Dr. Swagato Chatterjee from the VGSOM IIT, Kharagpur who is taking this course. We are in week seven, and we are discussing Recommendation Engine. So, in this video and probably in the next video also, we will discuss certain examples of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

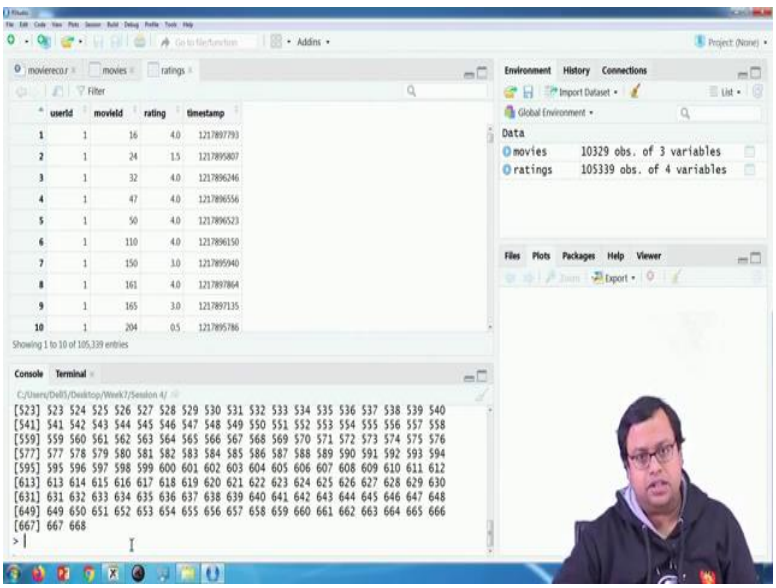
So, in this particular data set, here, we have movie ratings and where people have given ratings for movies. And I also have genre of the movies and I want to find out how I can recommend certain movies based on this genre and based on this, I would say interest towards those genre where data is being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

(Refer Slide Time: 1:10)




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
Showing 1 to 10 of 10,329 entries
Data
movies 10329 obs. of 3 variables
ratings 105339 obs. of 4 variables
> setwd("C:/Users/Dell5/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)

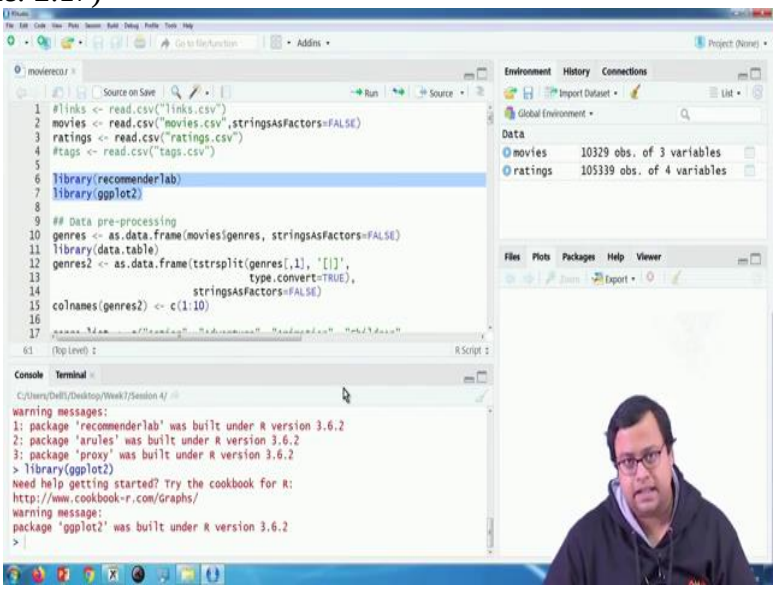

userId movieId rating timestamp
1 1 16 4.0 1217897793
2 1 24 1.5 1217895807
3 1 32 4.0 1217896246
4 1 47 4.0 1217896556
5 1 50 4.0 1217896523
6 1 110 4.0 1217896150
7 1 150 3.0 1217895940
8 1 161 4.0 1217897864
9 1 165 3.0 1217897135
10 1 204 0.5 1217895786
Showing 1 to 10 of 105,339 entries
Data
movies 10329 obs. of 3 variables
ratings 105339 obs. of 4 variables
> setwd("C:/Users/Dell5/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)

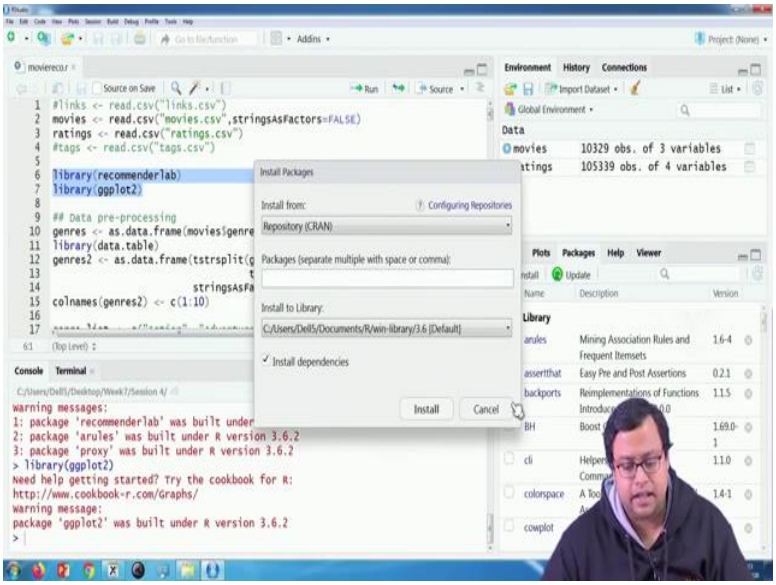

userId movieId rating timestamp
1 1 16 4.0 1217897793
2 1 24 1.5 1217895807
3 1 32 4.0 1217896246
4 1 47 4.0 1217896556
5 1 50 4.0 1217896523
6 1 110 4.0 1217896150
7 1 150 3.0 1217895940
8 1 161 4.0 1217897864
9 1 165 3.0 1217897135
10 1 204 0.5 1217895786
Showing 1 to 10 of 105,339 entries
Data
movies 10329 obs. of 3 variables
ratings 105339 obs. of 4 variables
[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
>

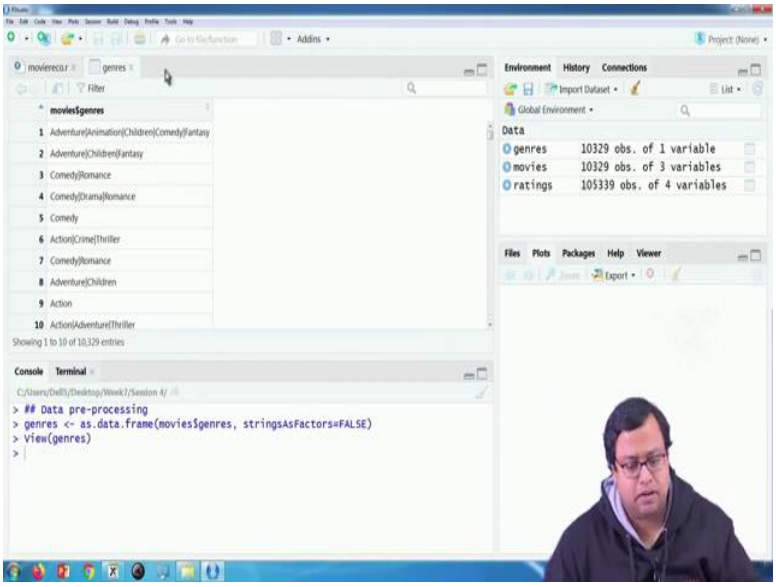
So here, the dataset, if I just read the dataset, the dataset looks like this, there are two datasets which we will merge. One is called movies dataset where the movie ID, the title of the movie, and the genres are written here. And then the ratings dataset, where basically around 1 lakh ratings are there, 1 lakh 5 thousand ratings are there, user ID, movie ID, rating and timestamp at what time point this guy has given the rating. So I will actually find out, so, there are basically if I am not wrong, ratings $ user ID, if I am trying to find out and then find out the unique of them. So, how many unique users are there?

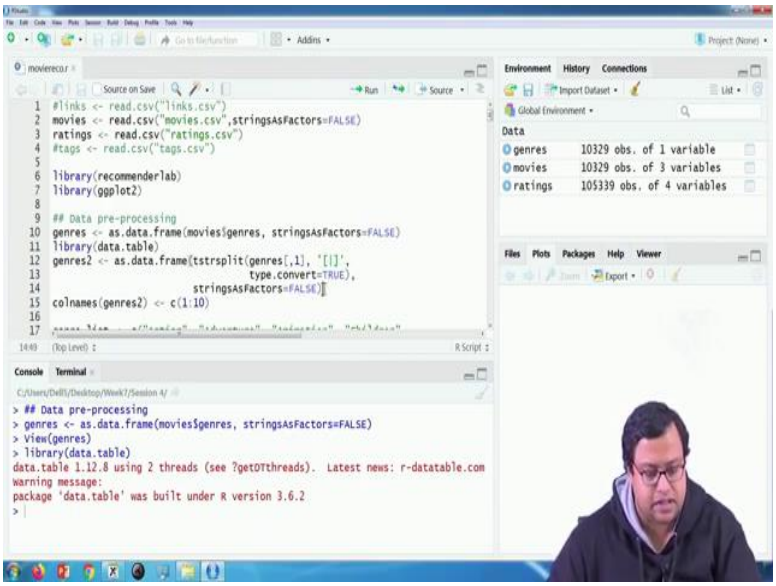
If I am not wrong, there are 668 unique users, yes. So, there 1 to 668 unique users are there and how many movies? Basically, there 10,329 movies. So, that is what I have. So, now, based on that, I have to find out which movie should be recommended.

(Refer Slide Time: 2:17)




#links <- read.csv("links.csv")
movies <- read.csv("movies.csv",stringsAsFactors=FALSE)
ratings <- read.csv("ratings.csv")
#tags <- read.csv("tags.csv")
library(recommenderlab)
library(ggplot2)
Data pre-processing
genres <- as.data.frame(movies$genre
library(data.table)
genres2 <- as.data.frame(tstrsplit(g
stringsAsFa
colnames(genres2) <- c(1:10)
Install Packages
Install from:
Configuring Repositories
Repository (CRAN)
Packages (separate multiple with space or comma):
Install to Library:
C:/Users/Dell5/Documents/R/win-library/3.6 [Default]
Install dependencies
Install
Cancel
warning messages:
1: package 'recommenderlab' was built under
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
Data
movies 10329 obs. of 3 variables
105339 obs. of 4 variables


moviesgenres
1 Adventure|Animation|Children|Comedy|Fantasy
2 Adventure|Children|Fantasy
3 Comedy|Romance
4 Comedy|Drama|Romance
5 Comedy
6 Action|Crime|Thriller
7 Comedy|Romance
8 Adventure|Children
9 Action
10 Action|Adventure|Thriller
Showing 1 to 10 of 10,329 entries
> ## Data pre-processing
> genres <- as.data.frame(moviesgenres, stringsAsFactors=FALSE)
> View(genres)
Data
genres 10329 obs. of 1 variable
movies 10329 obs. of 3 variables
ratings 105339 obs. of 4 variables

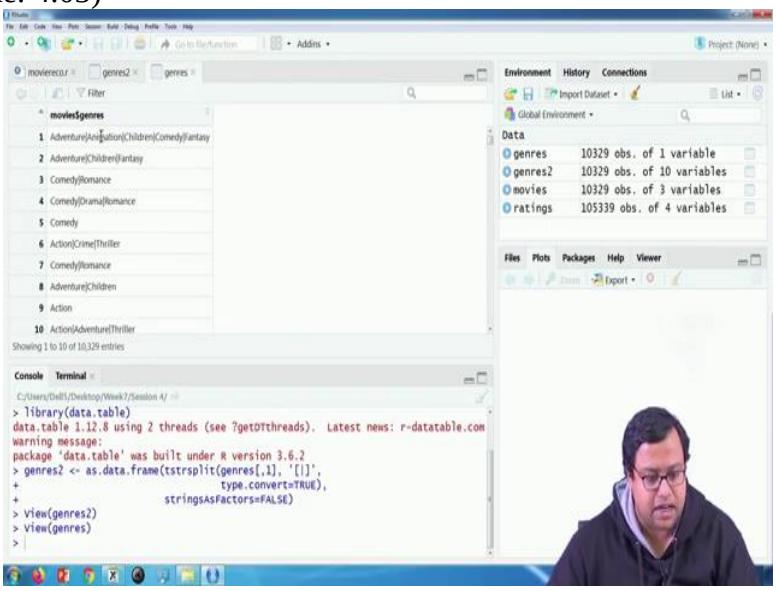

#links <- read.csv("links.csv")
movies <- read.csv("movies.csv",stringsAsFactors=FALSE)
ratings <- read.csv("ratings.csv")
#tags <- read.csv("tags.csv")
library(recommenderlab)
library(ggplot2)
Data pre-processing
genres <- as.data.frame(moviesgenres, stringsAsFactors=FALSE)
library(data.table)
genres2 <- as.data.frame(tstrsplit(genres[,1], '[|]',
type.convert=TRUE),
stringsAsFactors=FALSE)
colnames(genres2) <- c(1:10)
> ## Data pre-processing
> genres <- as.data.frame(moviesgenres, stringsAsFactors=FALSE)
> View(genres)
> 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
Data
genres 10329 obs. of 1 variable
movies 10329 obs. of 3 variables
ratings 105339 obs. of 4 variables

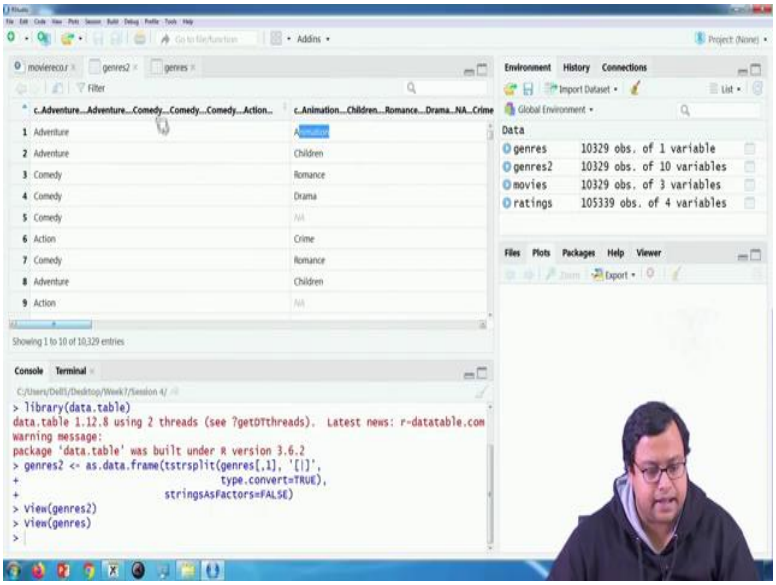
Now, if somebody has already rated a movie, then you will not see the movie anymore, then I have to find out his similar movie which he might like and likeness data, whether he will like or dislike a data, a movie will come from basically the rating that this guy has provided. So, I will ask for a library 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.

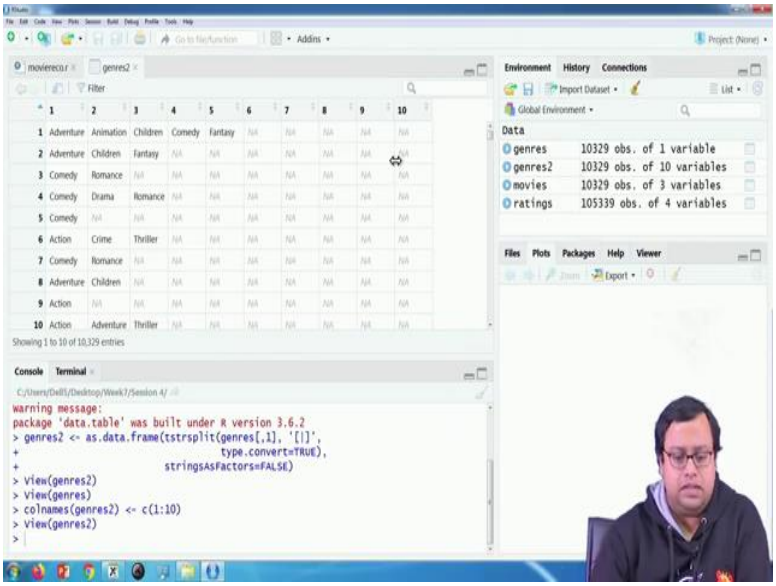
For bigger data, it helps so, recommender lab and ggplot2 is two libraries that I am calling, if you have not installed it, you have to first install it and then and how to install it? You can write install .packages and then the package name or you can go here and install and write the package name 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.

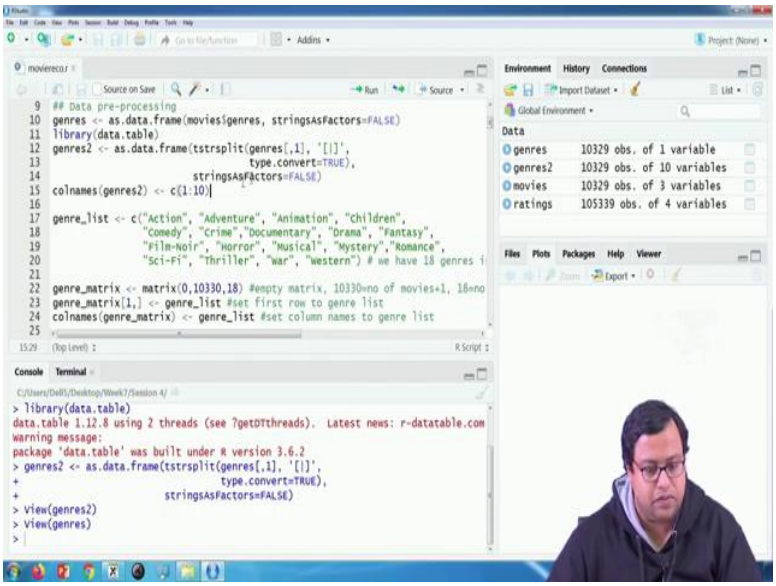
So, for that what I am doing is I am creating a genres column, so which is a genre column, only the genres of my dataset and then I am using a library called data.table. And then in with this library, I am breaking the genre with c, there is called string split. So, this string split will actually split 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.

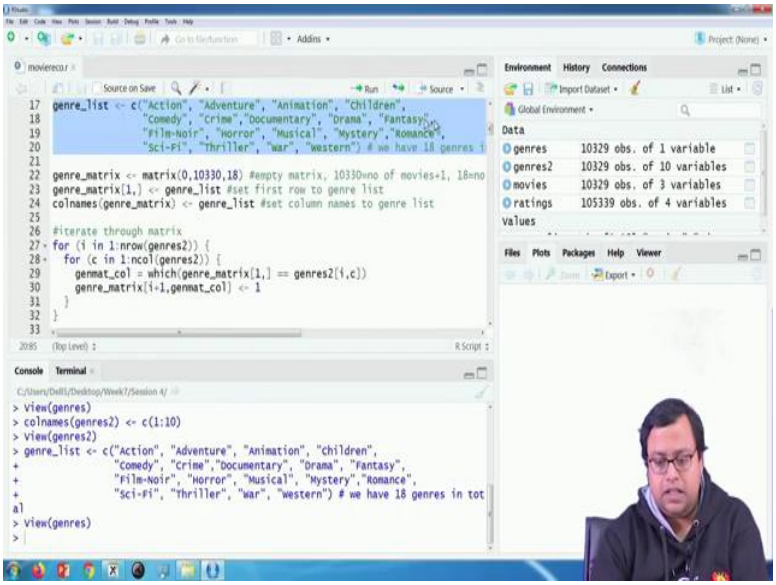
(Refer Slide Time: 4:03)




c..Adventure...Adventure...Comedy...Comedy...Comedy...Action...
c..Animation...Children...Romance...Drama...NA...Crime
1 Adventure Animation
2 Adventure Children
3 Comedy Romance
4 Comedy Drama
5 Comedy NA
6 Action Crime
7 Comedy Romance
8 Adventure Children
9 Action NA
Showing 1 to 10 of 10,329 entries
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)


1 2 3 4 5 6 7 8 9 10
1 Adventure Animation Children Comedy Fantasy
2 Adventure Children Fantasy
3 Comedy Romance
4 Comedy Drama Romance
5 Comedy
6 Action Crime Thriller
7 Comedy Romance
8 Adventure Children
9 Action
10 Action Adventure Thriller
Showing 1 to 10 of 10,329 entries
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
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)
> colnames(genres2) <- c(1:10)
> View(genres2)

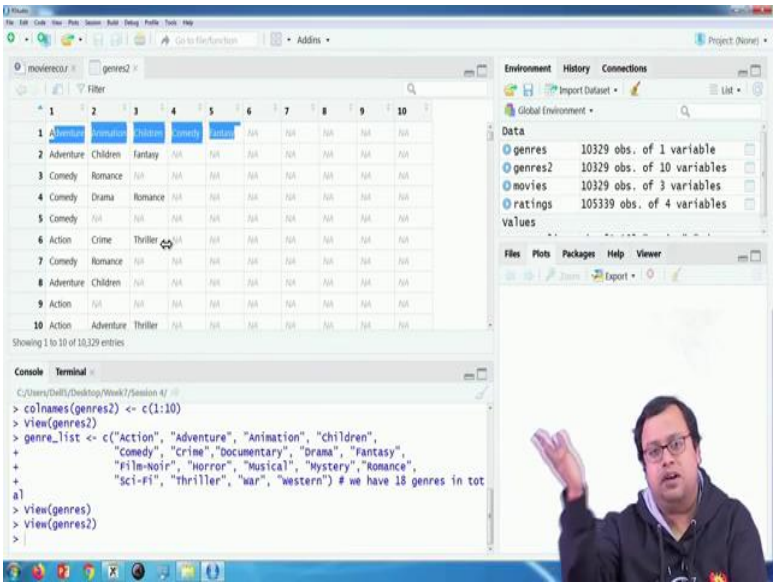

9 ## Data pre-processing
10 genres <- as.data.frame(movies$genres, stringsAsFactors=FALSE)
11 library(data.table)
12 genres2 <- as.data.frame(tstrsplit(genres[,1], '[|]',
13 type.convert=TRUE),
14 stringsAsFactors=FALSE)
15 colnames(genres2) <- c(1:10)
16
17 genre_list <- c("Action", "Adventure", "Animation", "Children",
18 "Comedy", "Crime","Documentary", "Drama", "Fantasy",
19 "Film-Noir", "Horror", "Musical", "Mystery","Romance",
20 "Sci-Fi", "Thriller", "War", "Western") # we have 18 genres i
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
15:29 (Top Level) R Script
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)

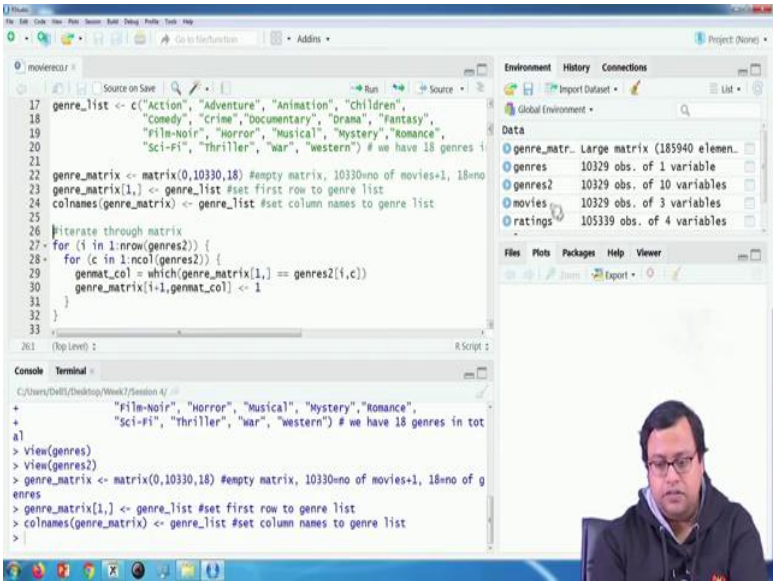


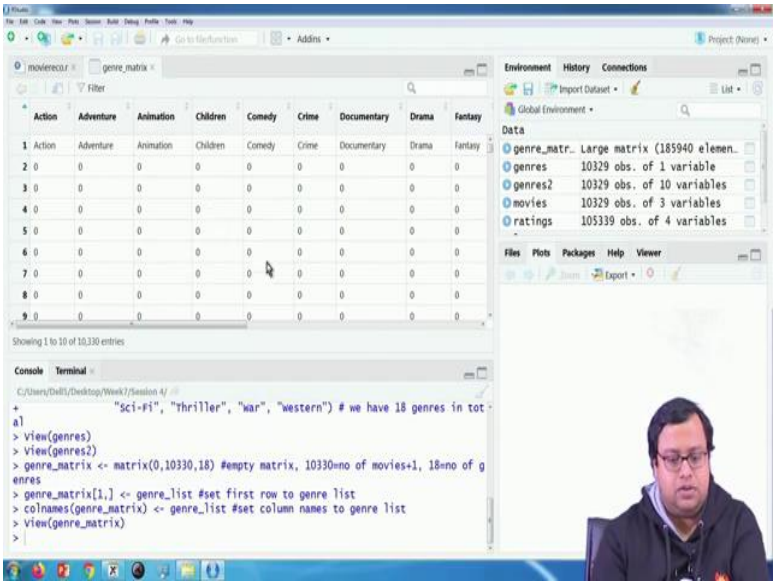
So, genre2 gets, what it does? It is actually breaking if you check genres if it is adventure, animation, children, comedy and fantasy, so adventure, animation, children, comedy and fantasy so, that is how it has broken and the rest of them are blind. So, we have created and if I just column names of 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)



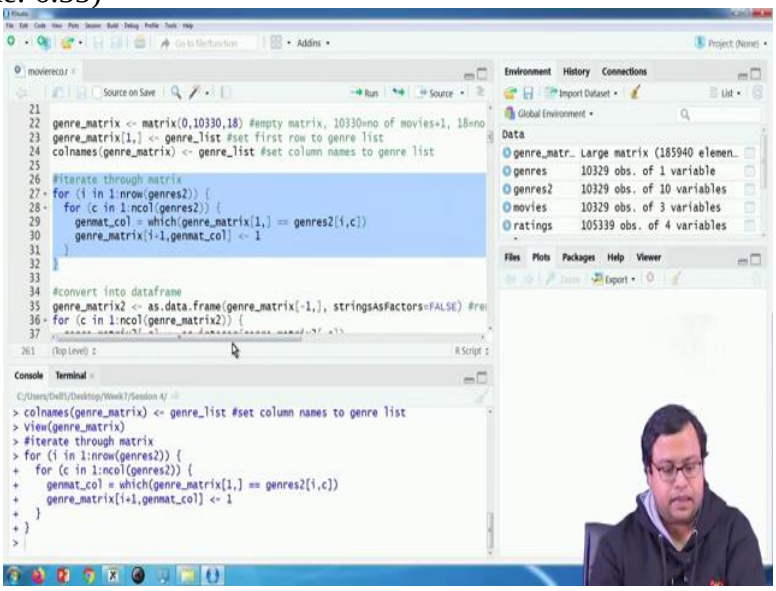


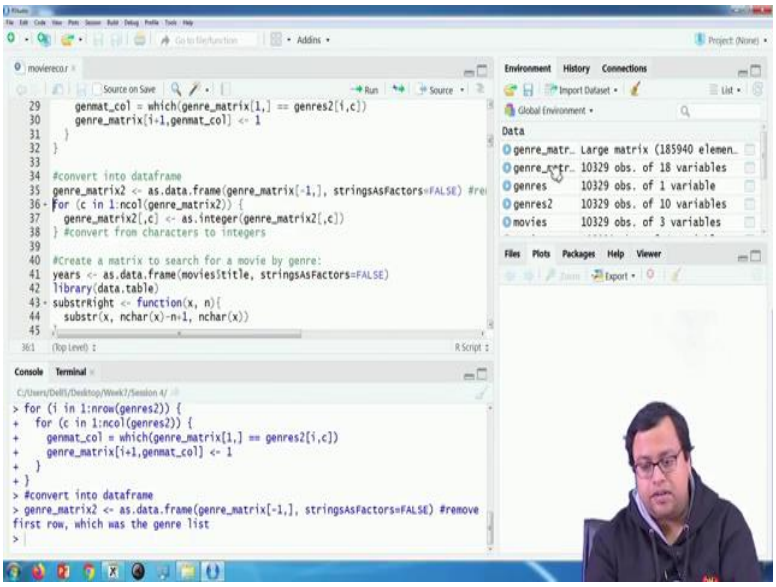


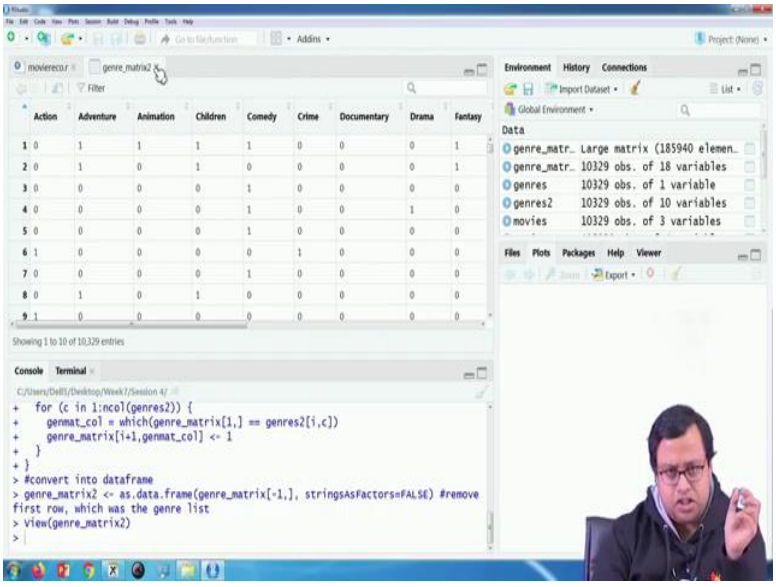
So, if this is the first particular movies genre, so then I will get a matrix in such a way such that that matrix will have number of movies is 10,329. So 10,329 rows and 18 columns, 18 columns each has one genre. And the value will be zero if that particular movie falls in that genre, and that data 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)

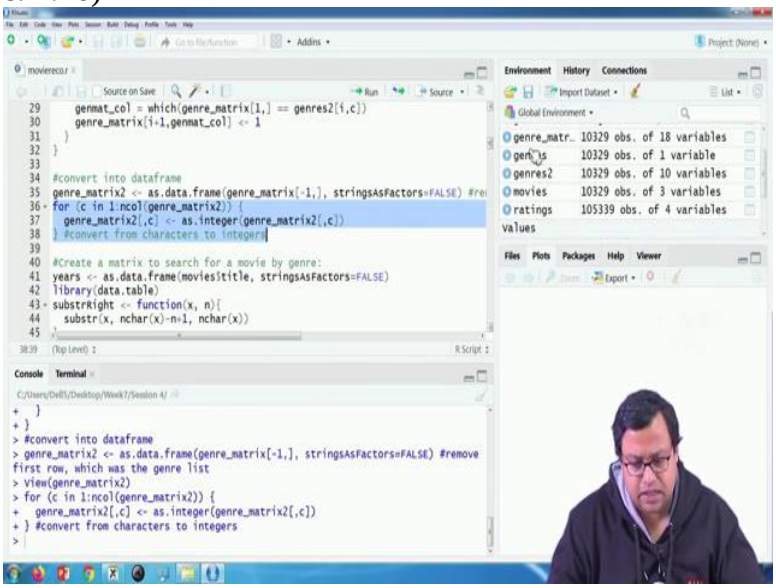


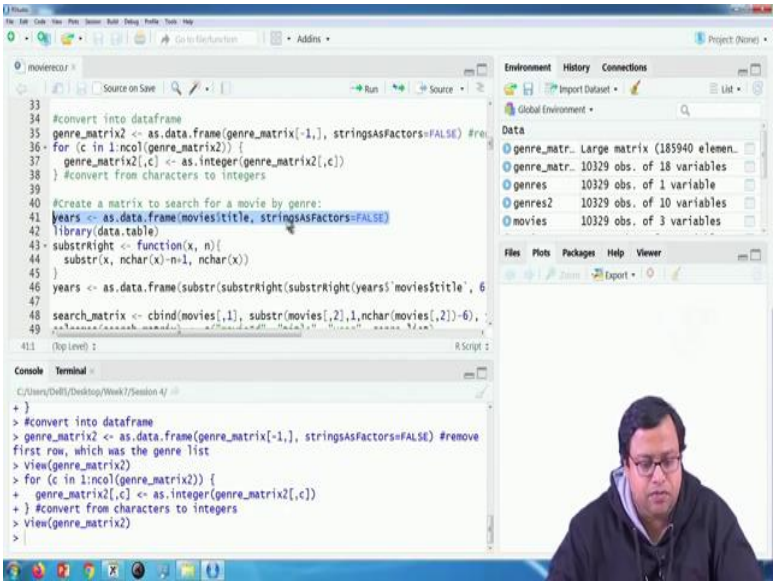


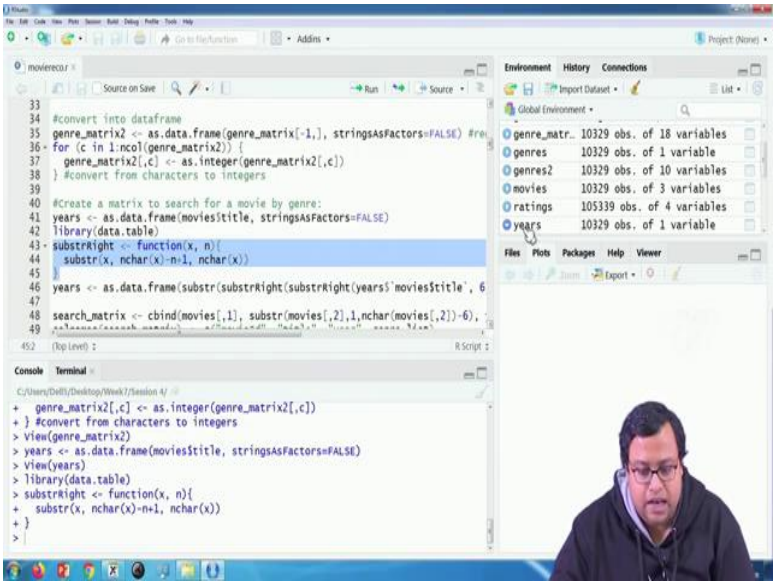


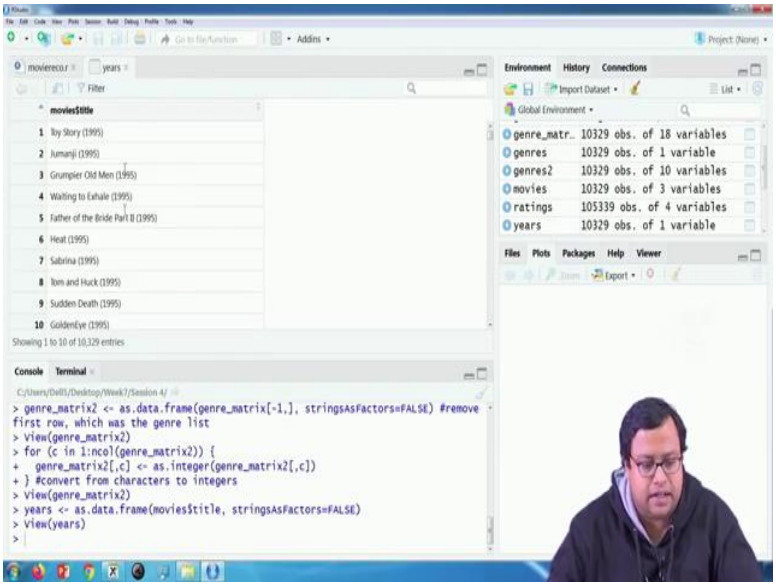
So, if I just run this thing quickly and then drop the first row, see first row I am dropping here, if I just drop the first row and what do I get in genre matrix2? I am getting genre matrix 2 like this. So, the first movie is action, animation, children, comedy, the second movie is adventure, children 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.

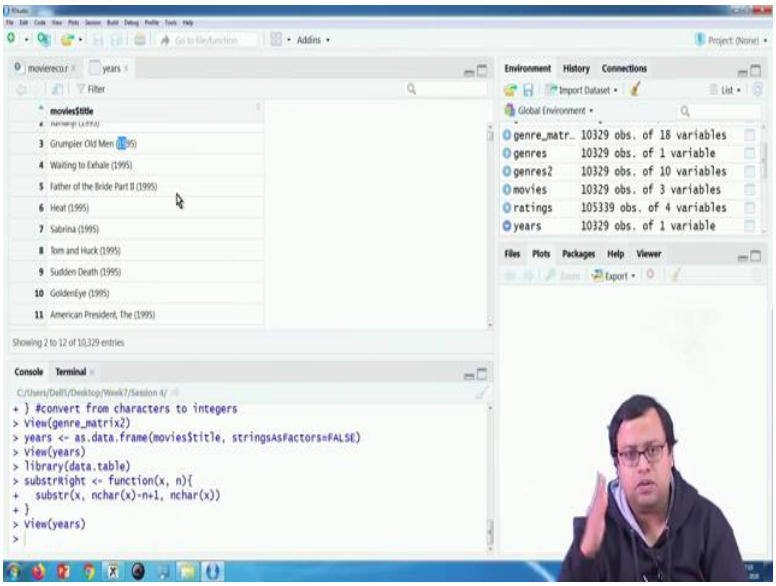
(Refer Slide Time: 7:16)






#convert into dataframe
genre_matrix2 <- as.data.frame(genre_matrix[-1,], stringsAsFactors=FALSE)
for (c in 1:ncol(genre_matrix2)) {
genre_matrix2[,c] <- as.integer(genre_matrix2[,c])
} #convert from characters to integers
#Create a matrix to search for a movie by genre:
years <- as.data.frame(movies$title, stringsAsFactors=FALSE)
library(data.table)
substrRight <- function(x, n){
substr(x, nchar(x)-n+1, nchar(x))
}
years <- as.data.frame(substr(substrRight(substrRight(years$`movies$title`, 6
search_matrix <- cbind(movies[,1], substr(movies[,2],1,nchar(movies[,2])-6),
+ genre_matrix2[,c] <- as.integer(genre_matrix2[,c])
+ } #convert from characters to integers
> View(genre_matrix2)
> years <- as.data.frame(movies$title, stringsAsFactors=FALSE)
> View(years)
> library(data.table)
> substrRight <- function(x, n){
+ substr(x, nchar(x)-n+1, nchar(x))
+ }
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


movies$title
1 Toy Story (1995)
2 Jumanji (1995)
3 Grumpier Old Men (1995)
4 Waiting to Exhale (1995)
5 Father of the Bride Part II (1995)
6 Heat (1995)
7 Sabrina (1995)
8 Tom and Huck (1995)
9 Sudden Death (1995)
10 GoldenEye (1995)
Showing 1 to 10 of 10,329 entries
> genre_matrix2 <- as.data.frame(genre_matrix[-1,], stringsAsFactors=FALSE) #remove first row, which was the genre list
> View(genre_matrix2)
> for (c in 1:ncol(genre_matrix2)) {
+ genre_matrix2[,c] <- as.integer(genre_matrix2[,c])
+ } #convert from characters to integers
> View(genre_matrix2)
> years <- as.data.frame(movies$title, stringsAsFactors=FALSE)
> View(years)

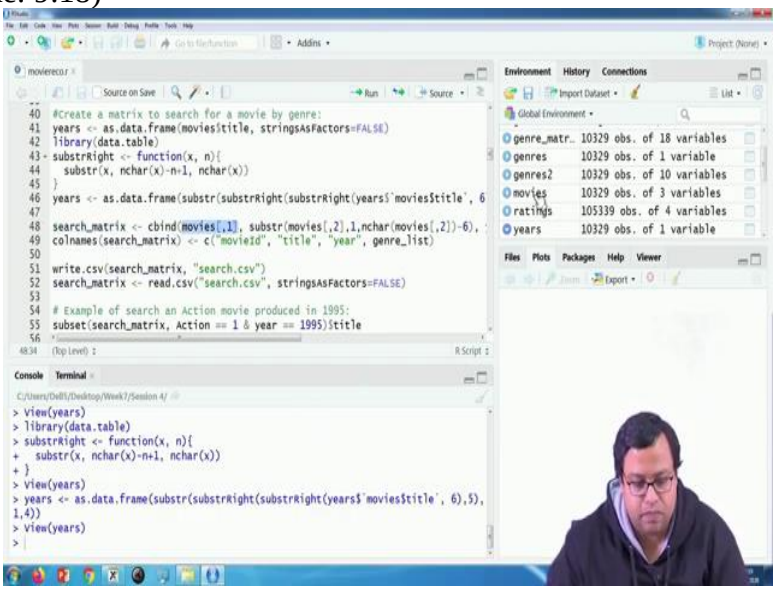

movies$title
3 Grumpier Old Men (1995)
4 Waiting to Exhale (1995)
5 Father of the Bride Part II (1995)
6 Heat (1995)
7 Sabrina (1995)
8 Tom and Huck (1995)
9 Sudden Death (1995)
10 GoldenEye (1995)
11 American President, The (1995)
Showing 2 to 12 of 10,329 entries
+ } #convert from characters to integers
> View(genre_matrix2)
> years <- as.data.frame(movies$title, stringsAsFactors=FALSE)
> View(years)
> library(data.table)
> substrRight <- function(x, n){
+ substr(x, nchar(x)-n+1, nchar(x))
+ }
> View(years)

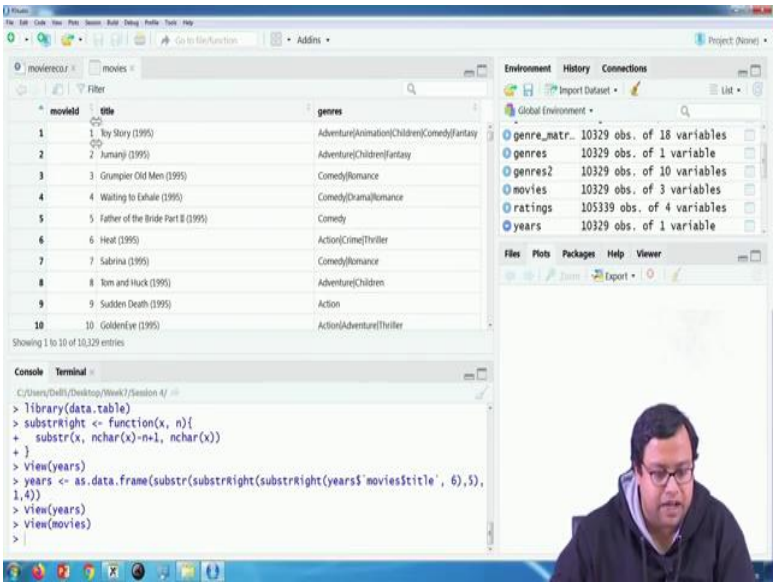
Now, next is for c in 1 to 18 n column genre matrix2 is 18, so, c is 1 to 18 the genre matrix for each column I am changing it to their integer values. So, if I just run this, is doing nothing but changing all values to their integer values. Now, what will I do? I will check, create a search for a movie 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.

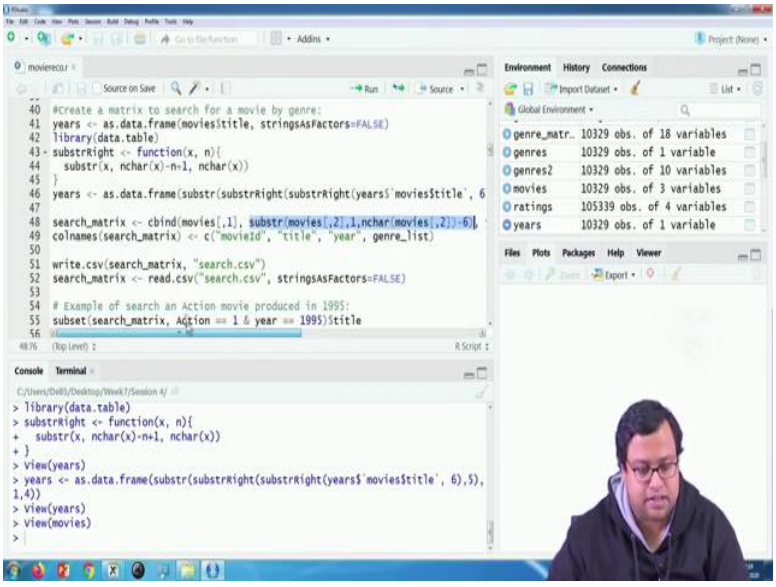
And then I am using library called data table, to what to do? To substring so, I am saying that substring write is a function, which will substring from left to right. So, this is what I am doing substring function and using the substring function from here, I will only take the years this 1995, this 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.

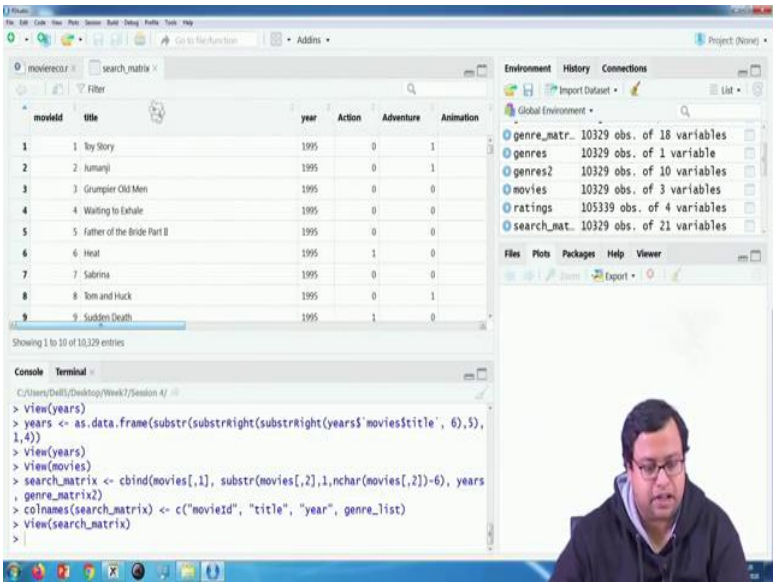
So years is equal to you see that 1 , 4 basically is something that it is doing, it is subtracting, creating a subset and that years is getting me all this year values of this particular thing. So again, you can do it in Excel if you want, but I am doing it here.

(Refer Slide Time: 9:18)



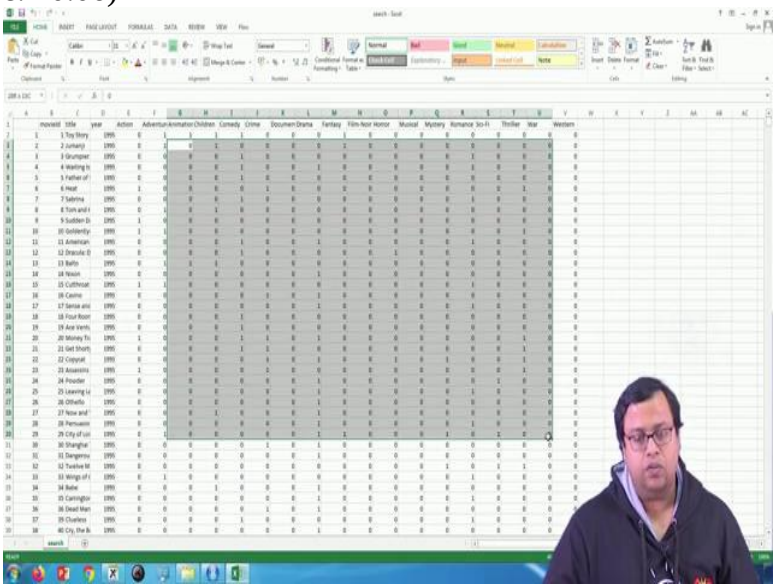


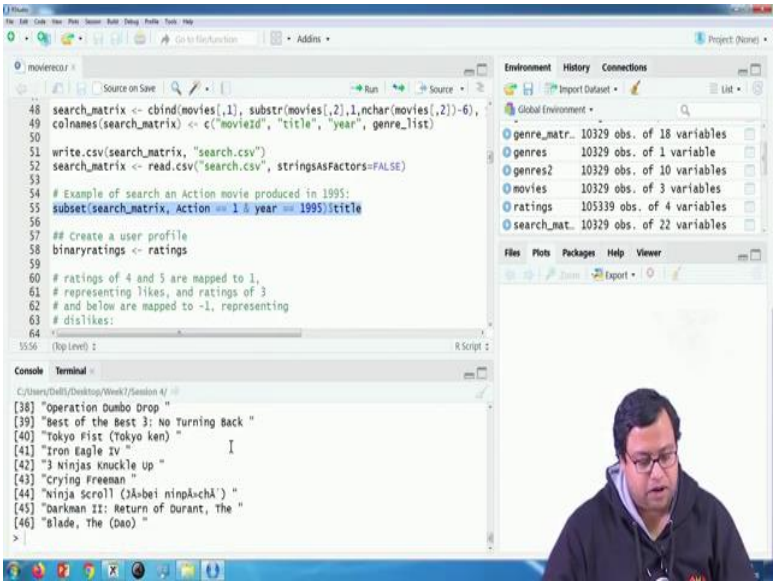


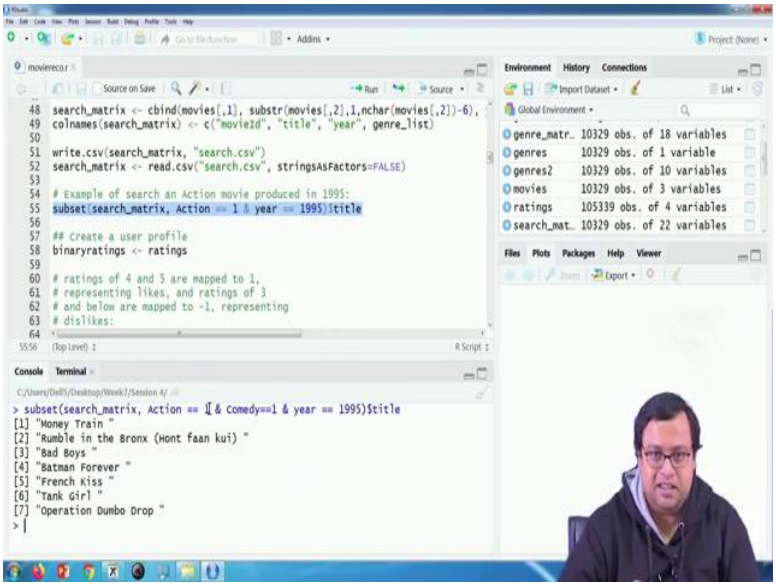


Next is I will create the same thing with a movie. So I am getting a search matrix, which is binding movies data first column. That means movie ID, next is substring of movies one, two, so the movie name, basically only the movie name, then the years and then the genre matrix. So, when I put this and 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)







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.

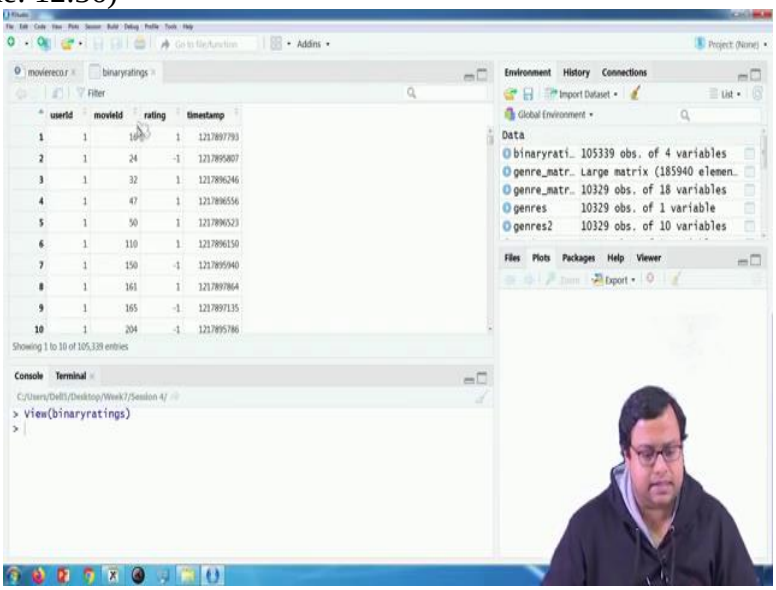
Now, what I will do next is I will create a binary rating. So, create a user profile is what is my next objective, so I am creating binary ratings anything 4 and 5 is high, 3, 2, 1 is low. So, that is what I am creating here so, binary rating is equal to ratings and then if the value is greater

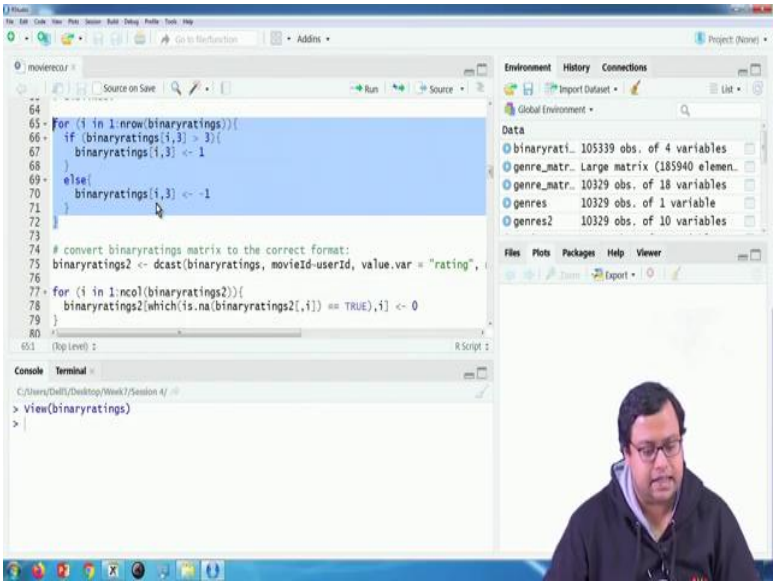
than 3 then put it 1, if it is smaller than 3 then - 1. So, that is what I just trying to do. Likes means 1, dislikes means - 1 so, it will run for some time. And let us see how much time it runs.

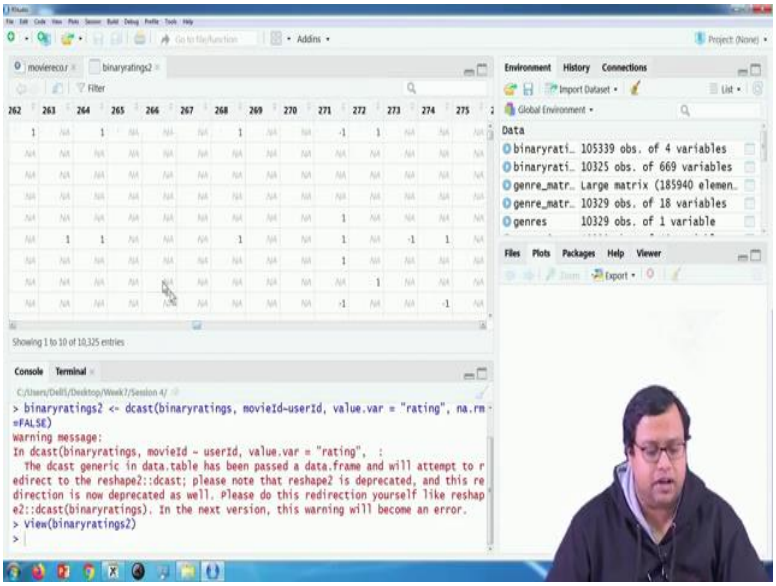
So, i in 1 to n row binary ratings, binary ratings is 1 lakh. So, it will take quite a bit of time to totally find out so, I will see it is now 28,000. And what I am trying to do here, basically it will go up to 1 lakh. What I am going to do is that sometimes it does not matter if there are multiple 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.

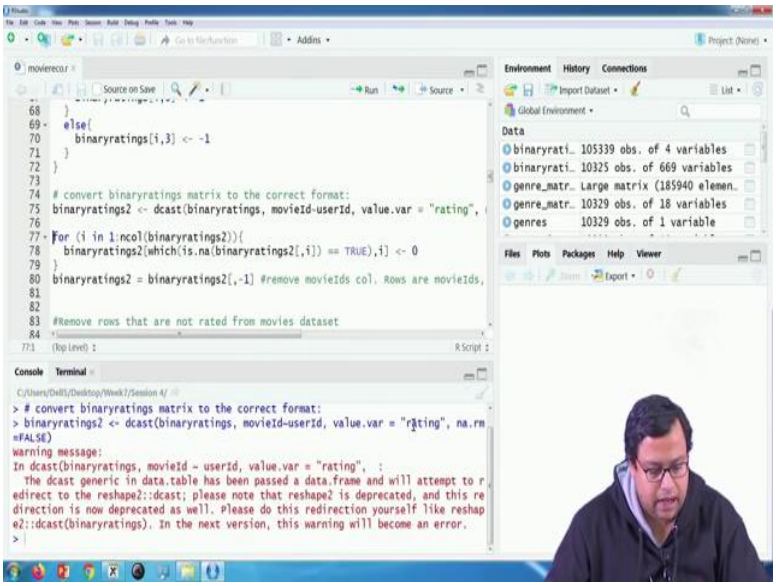
So, if it is positively skewed so, 1, 2, 3 is considered to be low and 4 and 5 is considered to be high and we are actually trying to do that we are in 98,000, 105000 will take okay. So, it is over so, I have created a binary rating. So, how does the binary ratings look like?

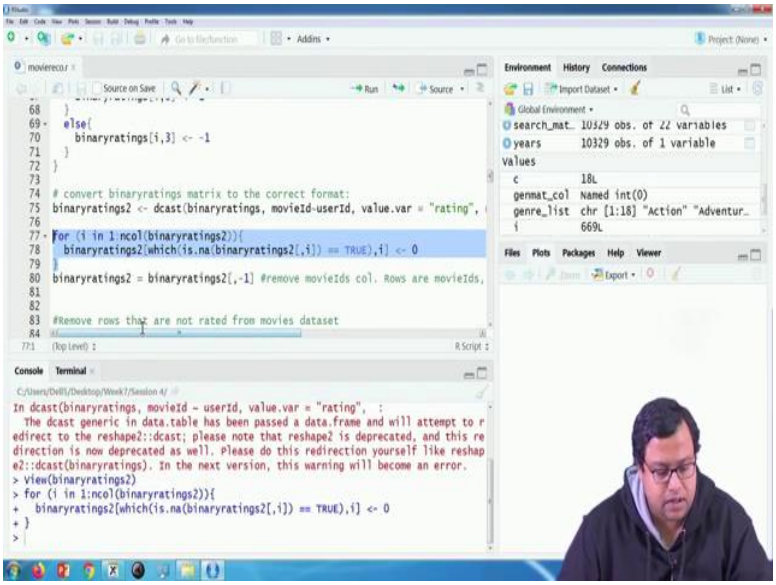
(Refer Slide Time: 12:36)




for (i in 1:nrow(binaryratings)){
if (binaryratings[i,3] > 3){
binaryratings[i,3] <- 1
}
else{
binaryratings[i,3] <- -1
}
}
# convert binaryratings matrix to the correct format:
binaryratings2 <- dcast(binaryratings, movieId~userId, value.var = "rating",
for (i in 1:ncol(binaryratings2)){
binaryratings2[which(is.na(binaryratings2[,i]) == TRUE),i] <- 0
}
> View(binaryratings)


Showing 1 to 10 of 10,325 entries
> 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 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)


}
else{
binaryratings[i,3] <- -1
}
}
# convert binaryratings matrix to the correct format:
binaryratings2 <- dcast(binaryratings, movieId~userId, value.var = "rating",
for (i in 1:ncol(binaryratings2)){
binaryratings2[which(is.na(binaryratings2[,i]) == TRUE),i] <- 0
}
binaryratings2 = binaryratings2[,-1] #remove movieIds col. Rows are movieIds,
#Remove rows that are not rated from movies dataset
> # 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 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.

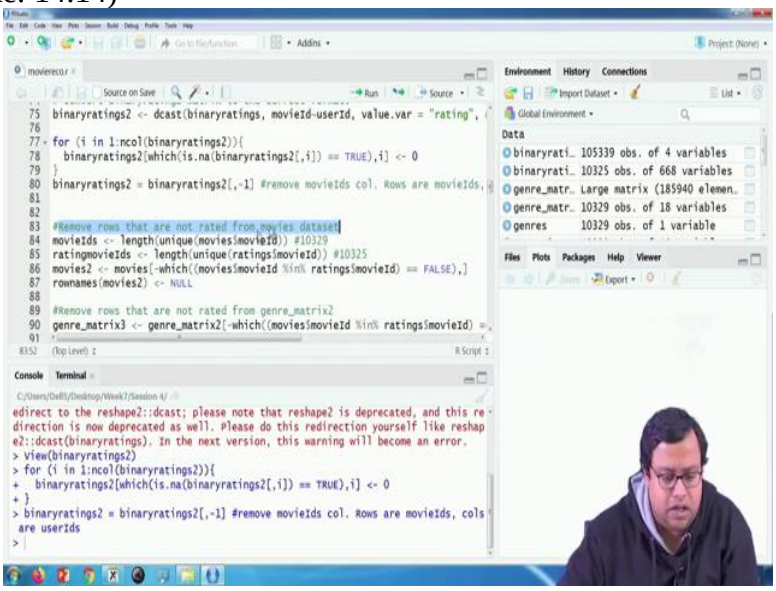


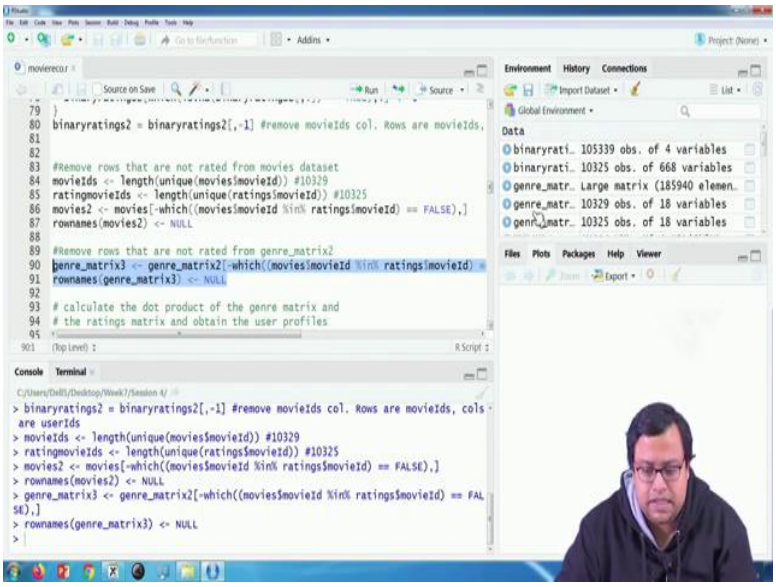
Binary ratings look like this, basic thing 1 or - 1 nothing else. So, this code is actually binary code you could have done a if else, simple if else would have taken much lesser time, anyways. So, then what I do is convert this binary rating so, what I will do is I will convert binary matrix with correct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

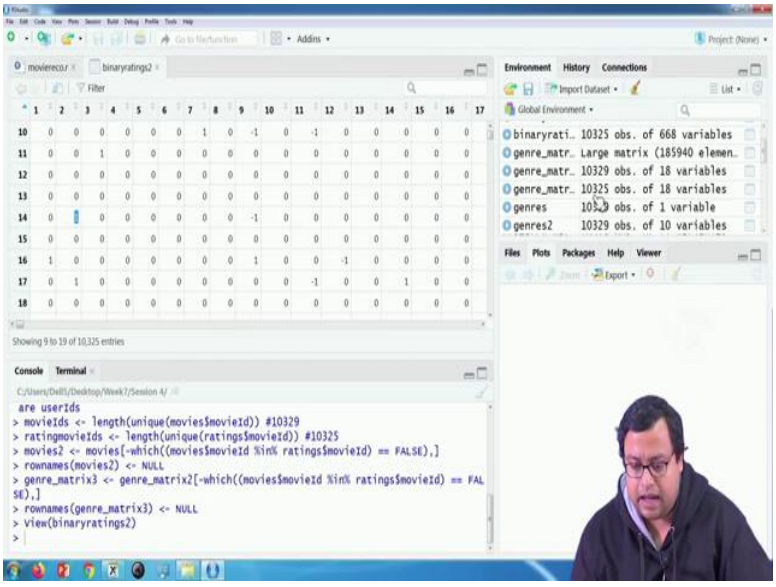
So, the format is like this binary ratings2 which I run and it will create a binary ratings2, which is the movie ID. And then correspondingly, there are lots of people 669 variables means 669 observations, all these NS means nobody has given review for that. And if they have given some review either 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.

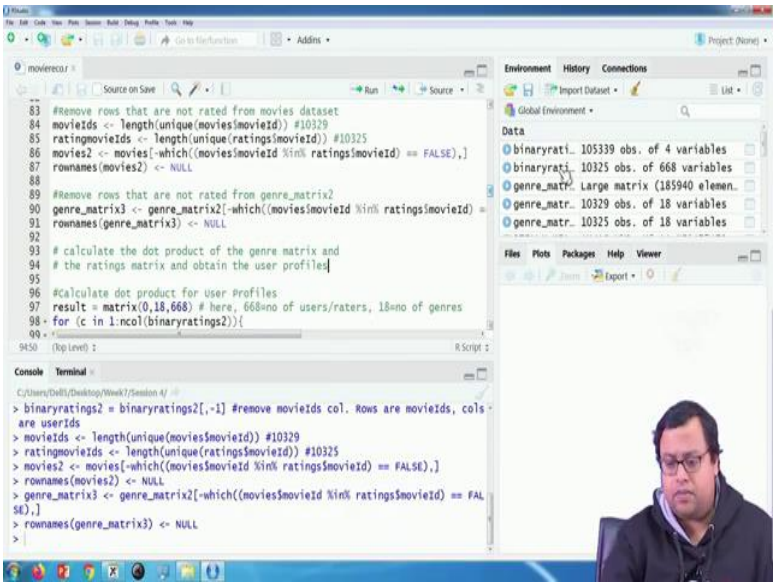
So, I am putting 0 for all the values where it is NA and then remove the movie ID column. So, that means I am getting up 10,325 observations, which is the movies, 668 observations, which are the users and 0 1 and - 1 this is the three values that are there, 0 means there was no rating, 1 means he liked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

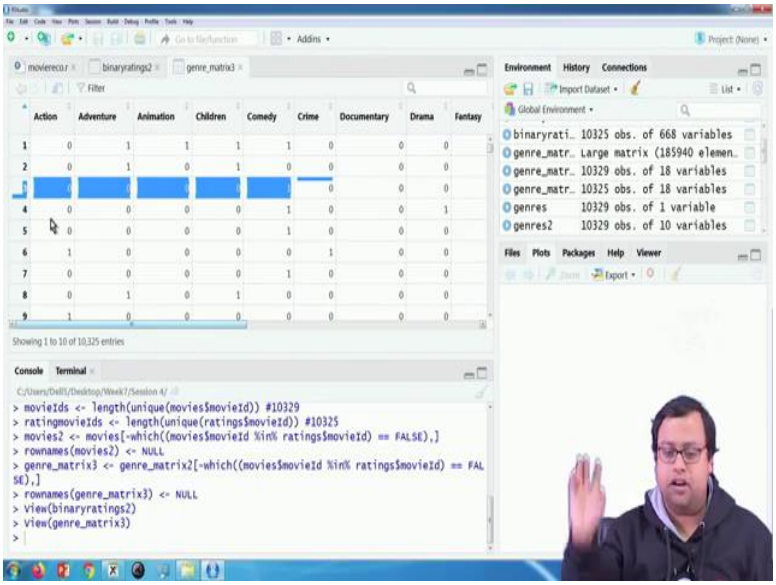
(Refer Slide Time: 14:14)











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

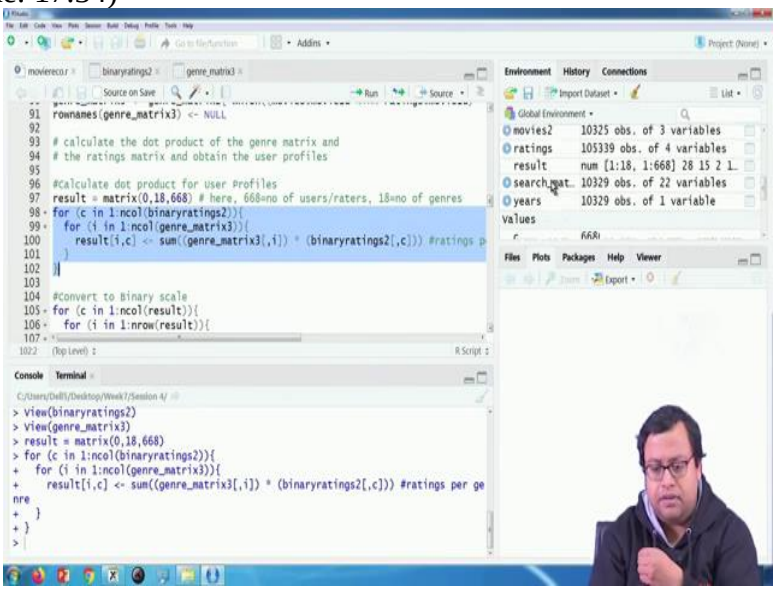
matrix and obtain the user profile. So, understand what is the genre matrix. Genre matrix is 10,325 of 18 genres, whether this movie is belonging to this genre, and then what is my rating matrix? Rating matrix is basically 10,325 of various people giving a rating.

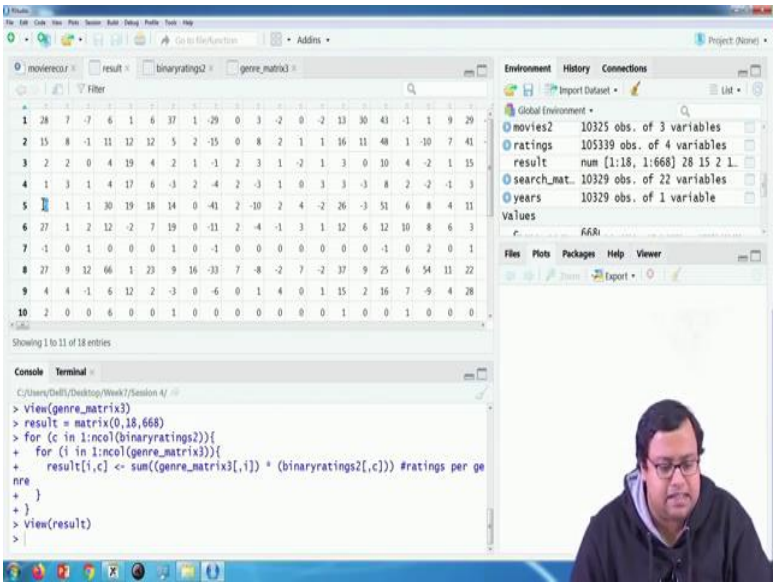
Now, if I want to find out that whether this user is fond of comedy, then what will I do? I have seen that this user, this 668 from this binary matrix, this one, if I just click on this one, let us see user number one. Let it come and then I will say, let us say user number one whether he so, user number 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 net 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)





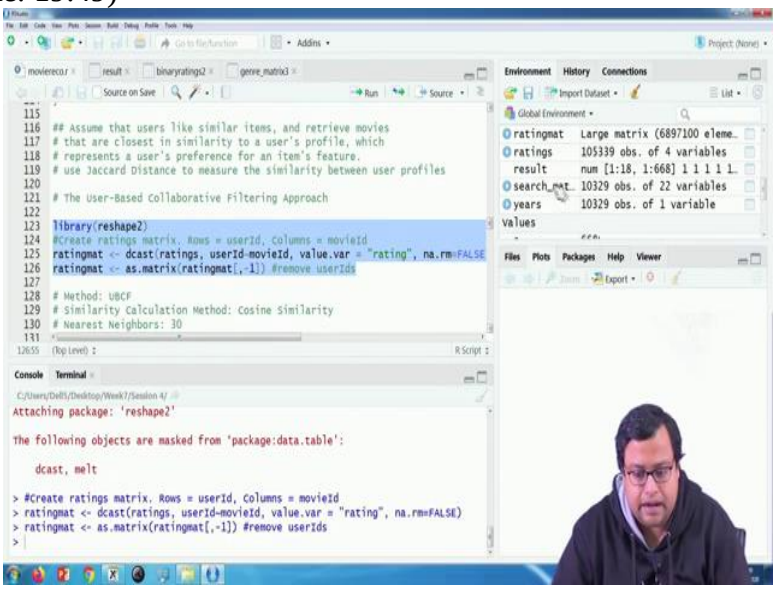
So that is what I am going to do here in the next set of calculations. So, I am creating a result which is 18 , 668, why 18? That means 18 rows, 668 columns, each column is one consumer or one user. And I am saying that whether he, basically a sum of 1's and - 1's, whether he has seen a movie in a particular 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.

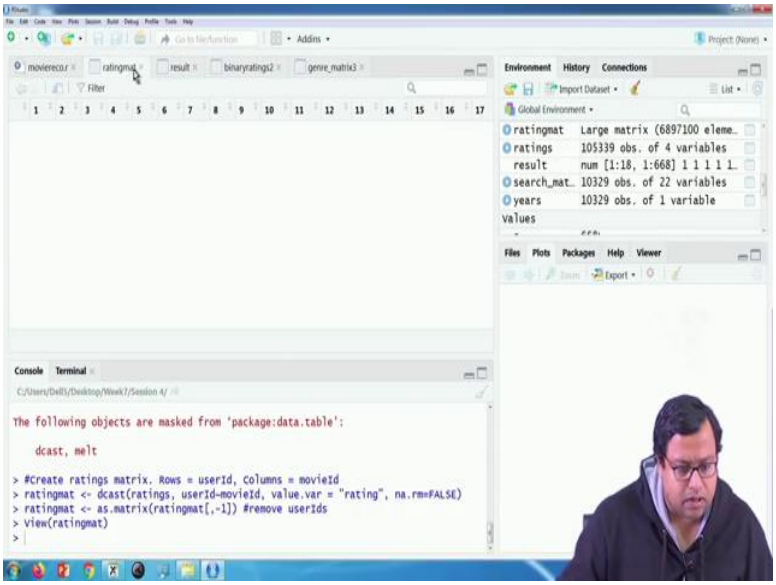
So now, if I just see this matrix carefully, what I get is user number 2 do not, user number 1 likes the first genre most 28, then sixth genre, the eighth genre and probably 16th genre. These are the genres that user number 1 likes, user number 2, probably this one, second genre, ninth genre. Now, if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

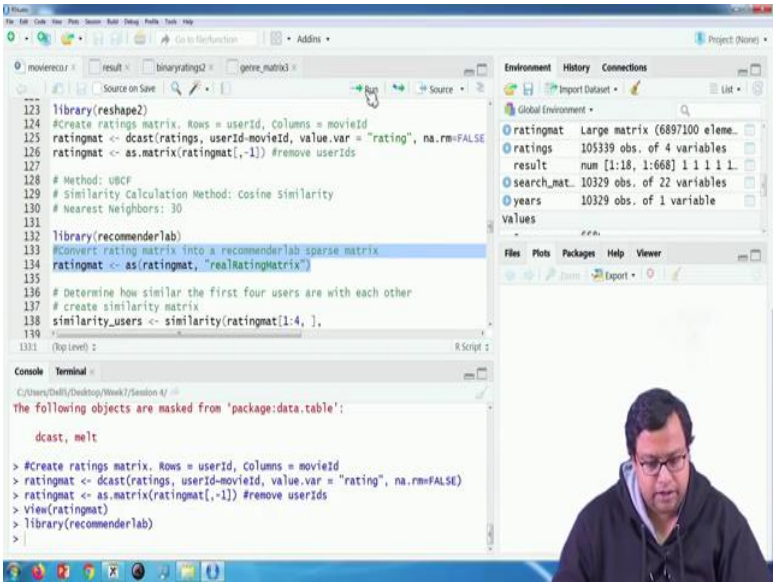
movie buff, he watches more movie, so the one which one he dislikes, likes a ninth is still much higher than the most liked one for the second user, which is 9 here. So he is 9 and this 9 has two different meaning. For the user 2 this 9 means that he likes the eighth genre most but he does not watch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

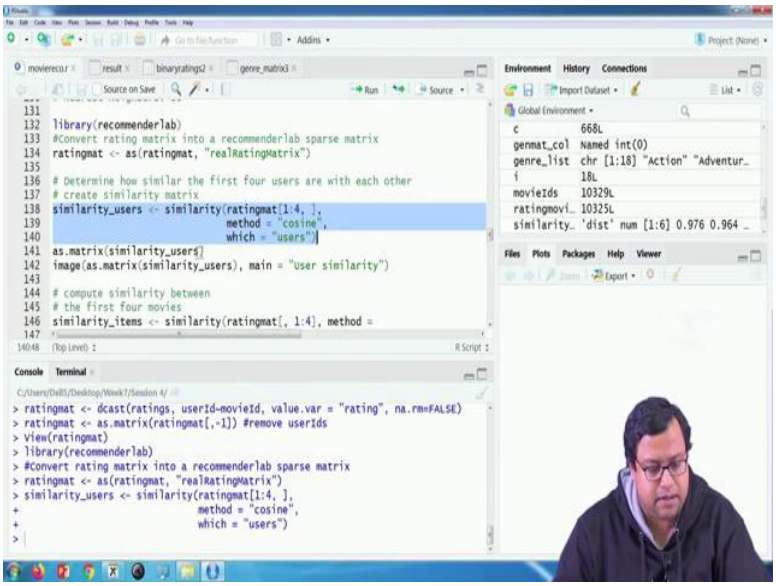
On the other hand, this 9 means that he watches movie very much. 9 rating is actually in comparison to 28, 27, 27 they are very small so, it is not his favorite genre, though the overall rating is 9. So, how to check that situation I have to normalize that. So, I am doing that, first of all convert the, 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.

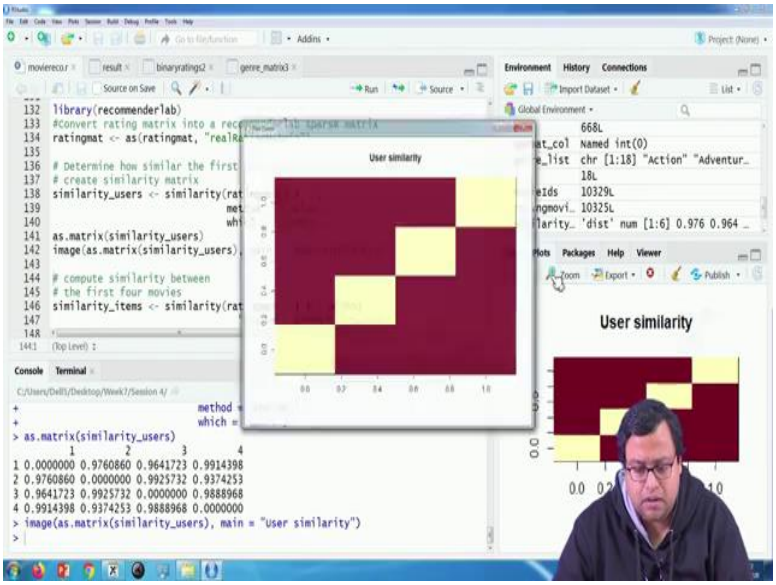
(Refer Slide Time: 19:49)

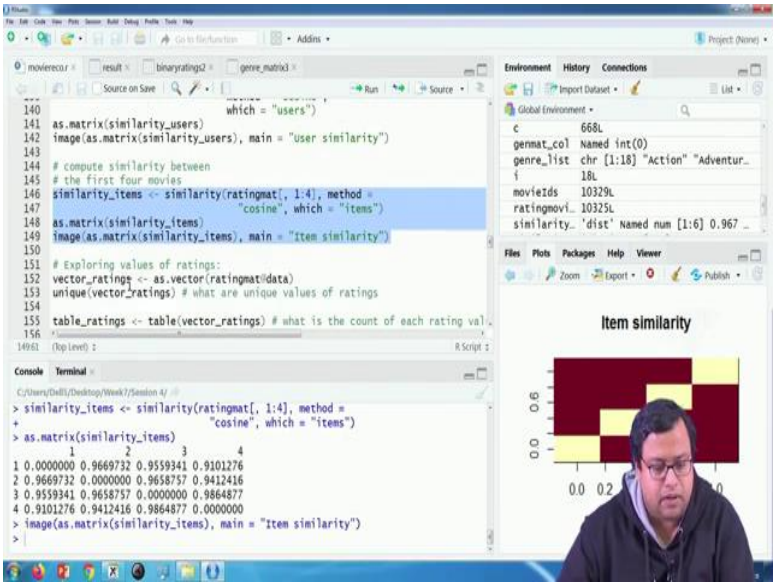



Environment History Connections
Import Dataset
Global Environment
ratingmat Large matrix (6897100 eleme...
ratings 105339 obs. of 4 variables
result num [1:18, 1:668] 1 1 1 1 1...
search_mat... 10329 obs. of 22 variables
years 10329 obs. of 1 variable
Values
Files Plots Packages Help Viewer
Export
Console Terminal
C:/Users/Dell5/Desktop/Week7/Session 4/
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
> View(ratingmat)


Source on Save
Run
Source
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 library(recommenderlab)
133 #Convert rating matrix into a recommenderlab sparse matrix
134 ratingmat <- as(ratingmat, "realRatingMatrix")
135
136 # Determine how similar the first four users are with each other
137 # create similarity matrix
138 similarity_users <- similarity(ratingmat[1:4,],
(Top Level)
R Script
Environment History Connections
Import Dataset
Global Environment
ratingmat Large matrix (6897100 eleme...
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
Values
Files Plots Packages Help Viewer
Export
Console Terminal
C:/Users/Dell5/Desktop/Week7/Session 4/
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
> View(ratingmat)
> library(recommenderlab)


Source on Save
Run
Source
131
132 library(recommenderlab)
133 #Convert rating matrix into a recommenderlab sparse matrix
134 ratingmat <- as(ratingmat, "realRatingMatrix")
135
136 # Determine how similar the first four users are with each other
137 # create similarity matrix
138 similarity_users <- similarity(ratingmat[1:4,],
139 method = "cosine",
140 which = "users")
141 as.matrix(similarity_users)
142 image(as.matrix(similarity_users), main = "User similarity")
143
144 # compute similarity between
145 # the first four movies
146 similarity_items <- similarity(ratingmat[, 1:4], method =
147
140:48 (Top Level)
R Script
Environment History Connections
Import Dataset
Global Environment
c 668L
genmat_col Named int(0)
genre_list chr [1:18] "Action" "Adventur...
i 18L
movieIds 10329L
ratingmovi... 10325L
similarity... 'dist' num [1:6] 0.976 0.964 ...
Files Plots Packages Help Viewer
Export
Console Terminal
C:/Users/Dell5/Desktop/Week7/Session 4/
> ratingmat <- dcast(ratings, userId~movieId, value.var = "rating", na.rm=FALSE)
> ratingmat <- as.matrix(ratingmat[,-1]) #remove userIds
> View(ratingmat)
> library(recommenderlab)
> #Convert rating matrix into a recommenderlab sparse matrix
> ratingmat <- as(ratingmat, "realRatingMatrix")
> similarity_users <- similarity(ratingmat[1:4,],
+ method = "cosine",
+ which = "users")





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.

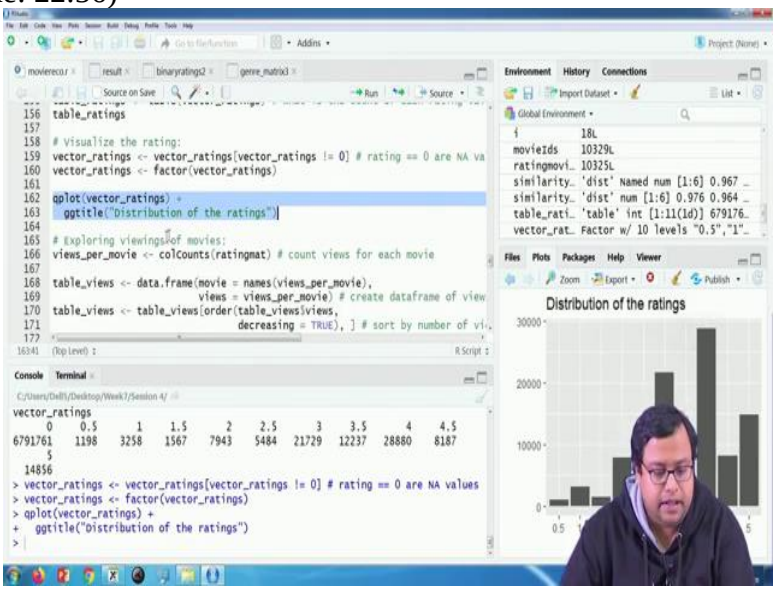
So, these are basically specific to this particular library you have to use this, you have no choice and then the similarity, basically similarity of the users is similarity rating mat 1 to 4, method is cosine which is user so, I am doing a user to users similarity matrix and if I try to find out a similarity 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.

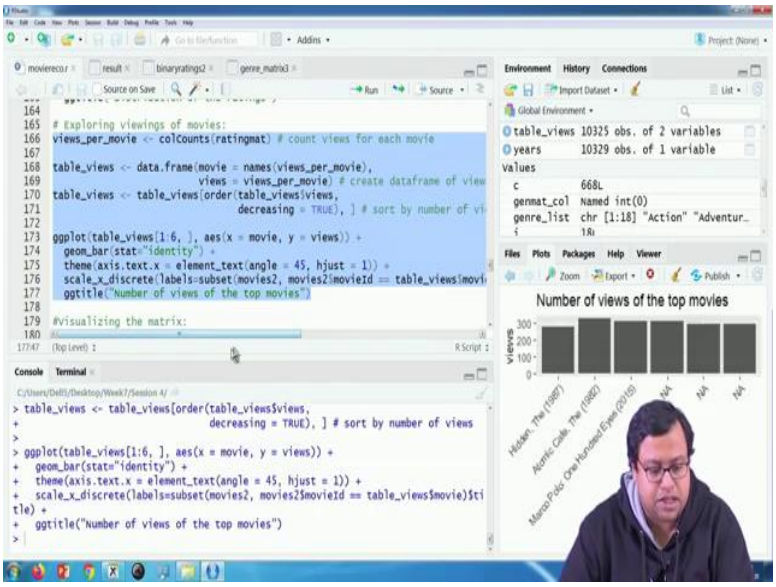
So, the yellowish it is the better, that is the image that we are getting so, this one is coming as to be good we have check the red colors again once more.

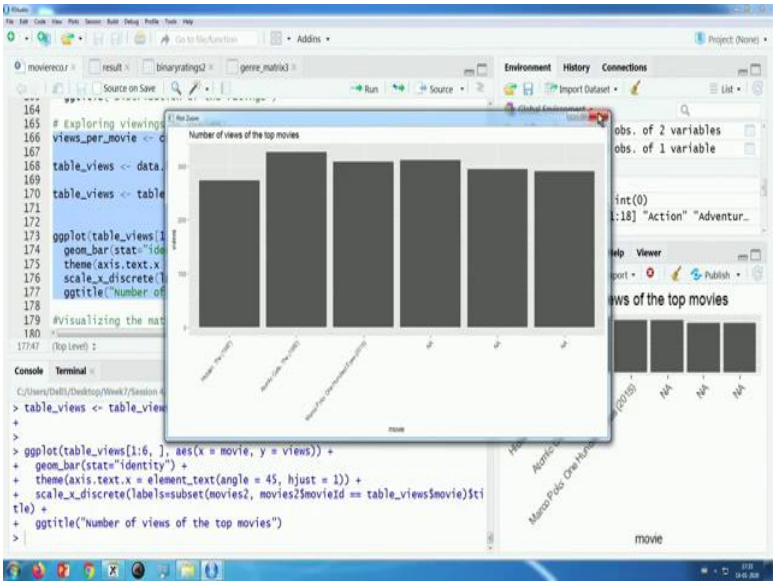
Then I can also find out the, compute similarity between the first four movies so, I can do it for the movie by movie similarity also so, for the first four movie if I try to find out the similarity, this is the similarity, see 0.9, 0.95, 0.91 so, the first four movies are very similar probably with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

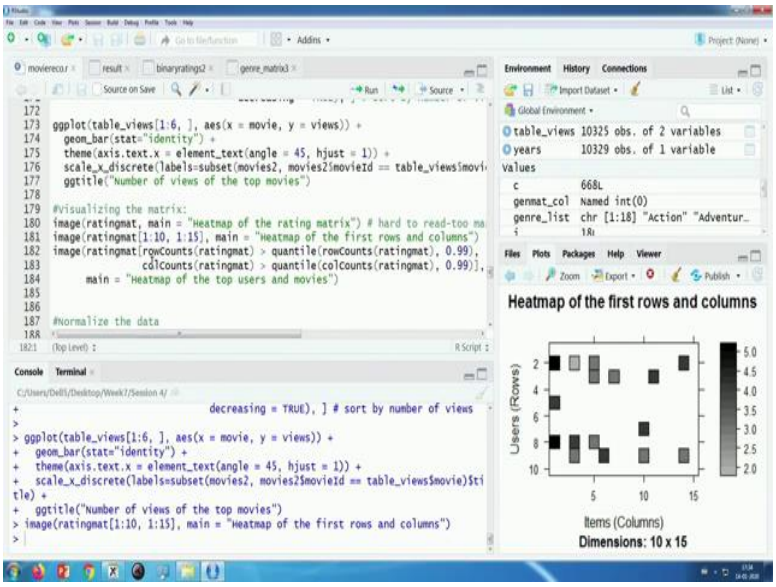
Now, explore the value for ratings so, this is where I am actually trying to find out the ratings, the value of ratings so, vector ratings is equal to as vector, rating mat $ data sorry, at the rate data so, this is a different format for sparse matrix and unique of vector rating. These are the various 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)





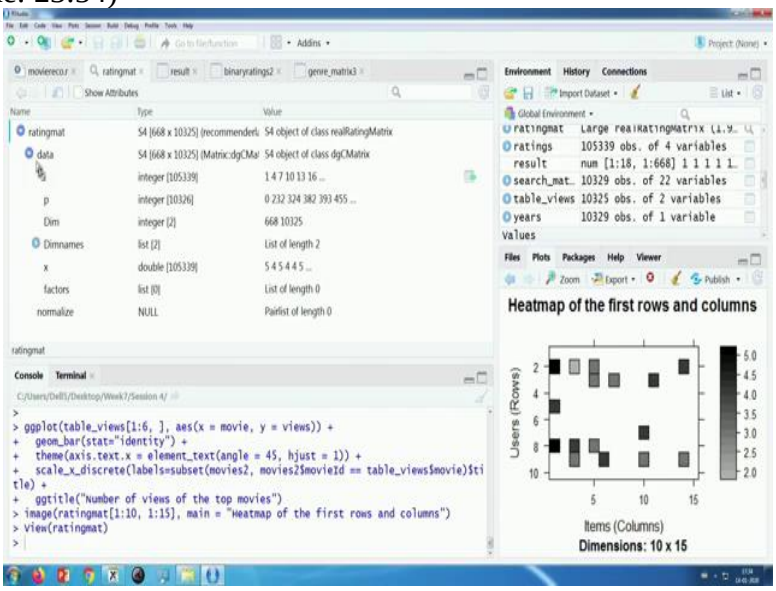


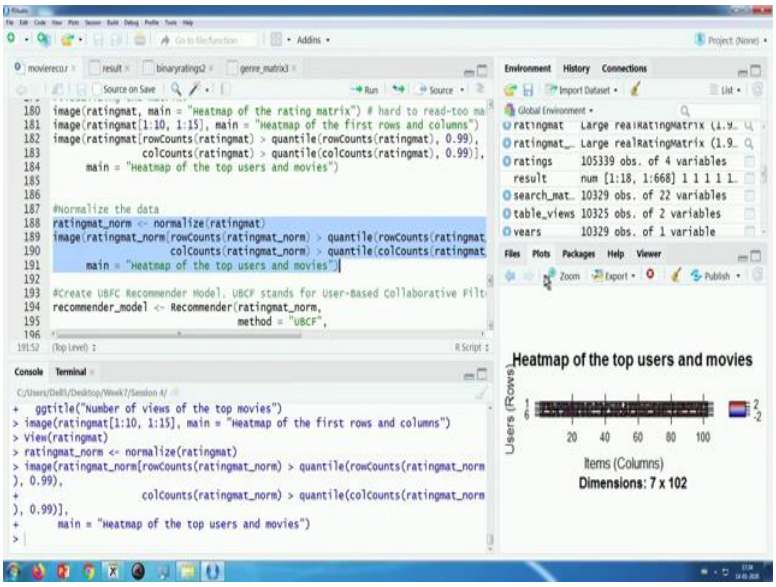


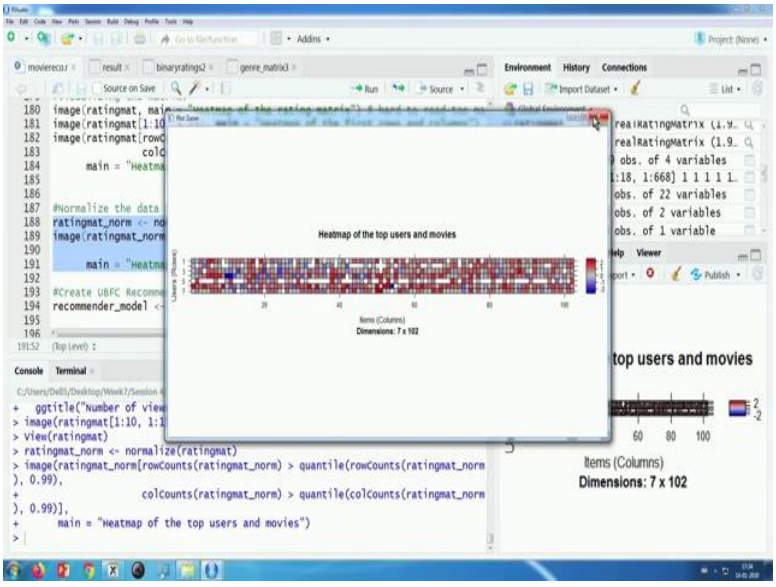
Now, vector ratings ≠ 0 aI will only take why it is ≠ zero? Because zeros are NA's if you remember, so, I am taking that and making them factor and then if I am plotting them by chance, it is basically a histogram plot that how many different kind of ratings that I am getting here. Then if I explore the viewing behavior with a similar way, I will get, I will not spend time on this, we will just check it, I get a viewing behavior of the top five movies also so, these are the top five movies viewing behavior.

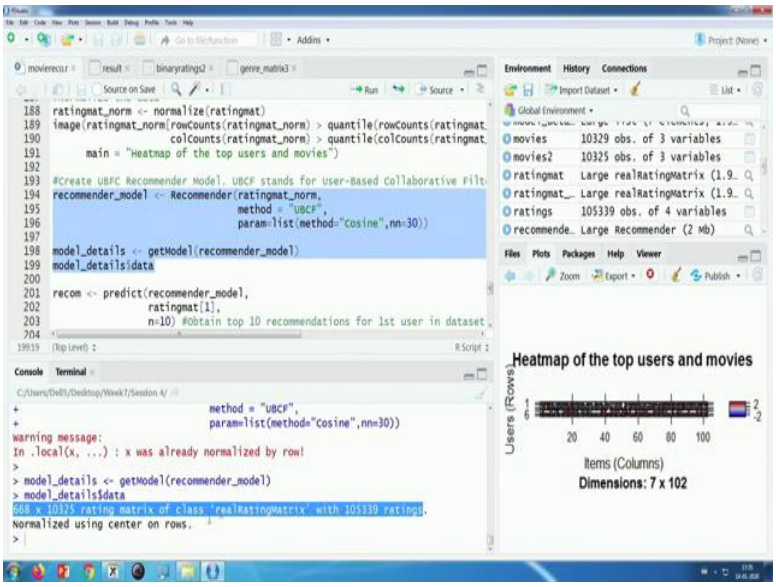
And then if I visualize the matrix, I should visualize this one, the heat map so, the heat map of the first rows and columns so, this is the user rows and this is the item columns. The darker it is, the more it is preference towards that based on the dataset that we have got. Now, what I will do is I will that rating mat if you remember, the rating mat, this is what I am coming back once more the rating mat dataset.

(Refer Slide Time: 23:54)





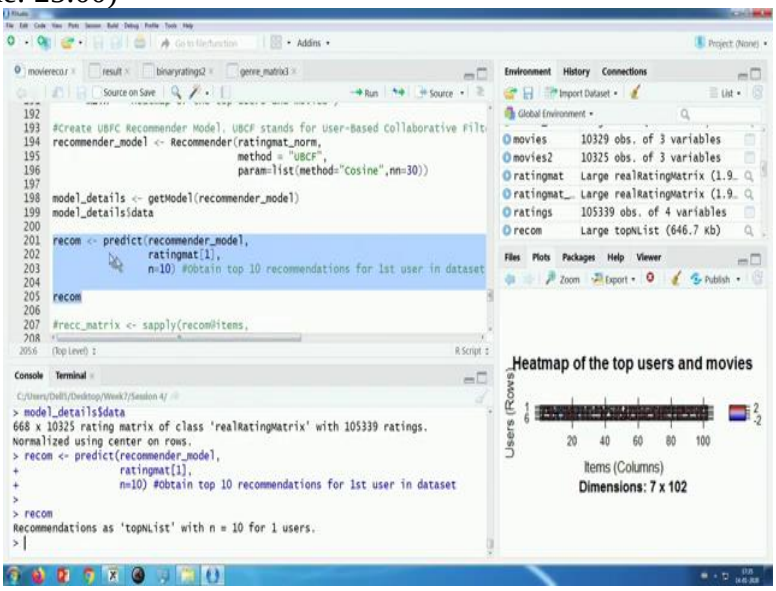


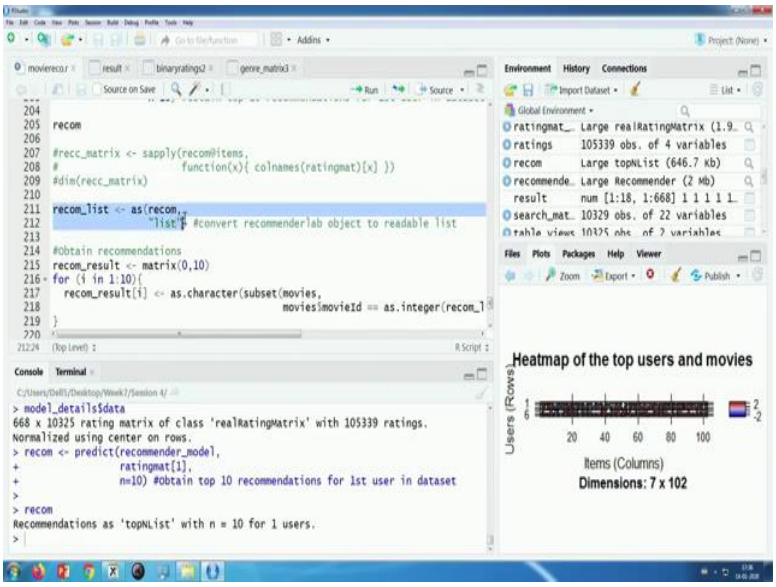


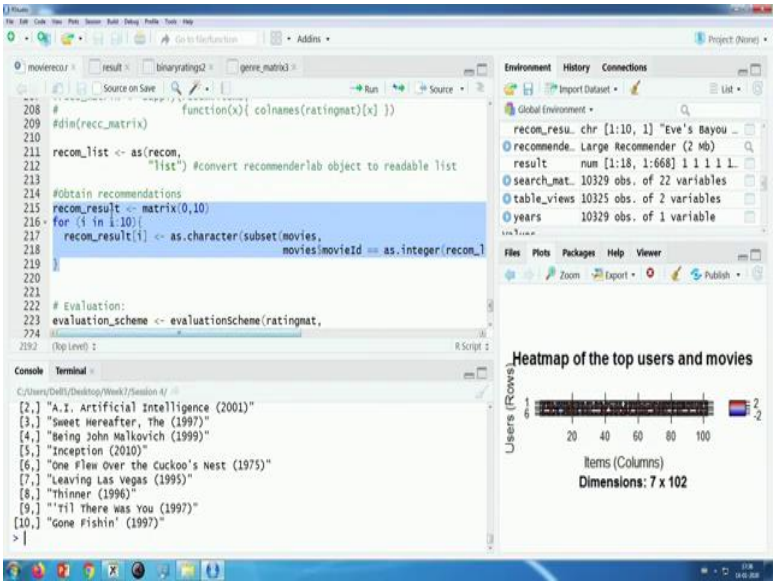
This is actually the one that I will use in my UBCF the recommender lab library. So, I am normalizing it first and after normalization this is the values that you are getting, the item columns and the user rows. And all these 1's are basically the heat map, what is a similarity level 2 to - 2, various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

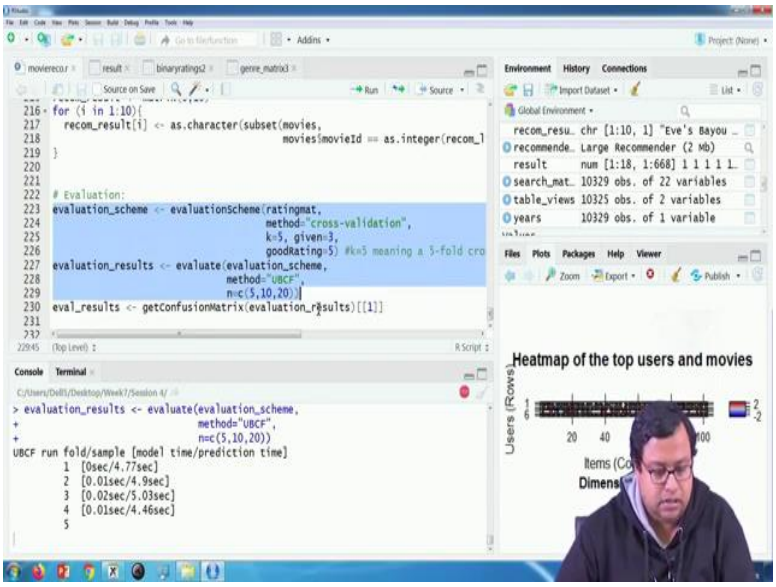
And then I use UBCF, UBCF means? User based collaborative filtering, and nearest number = 30, cosine is equal to, method is equal to cosine, you can change this cosine to correlation and etcetera. So, that is how you get the model and you find out the details of the data. So, the model details of the 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)







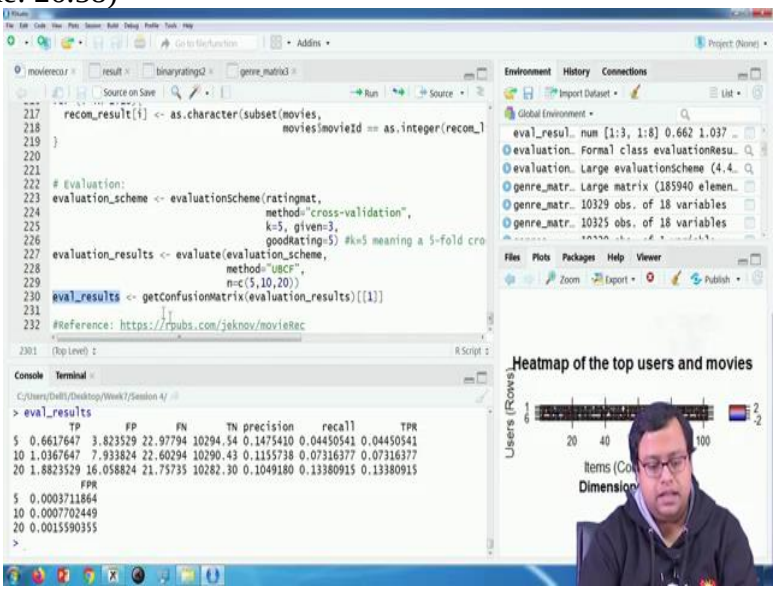


Now, if you want to find out a recommendation, the top 10 recommendation. So, for the first person I am taking rating mat 1, if you change this to 2 or 1 to 10, you will get the recommendations for all the 10 guys. So, recommendations for 10, it is the top 10 list for one user is something that is saved 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)



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.