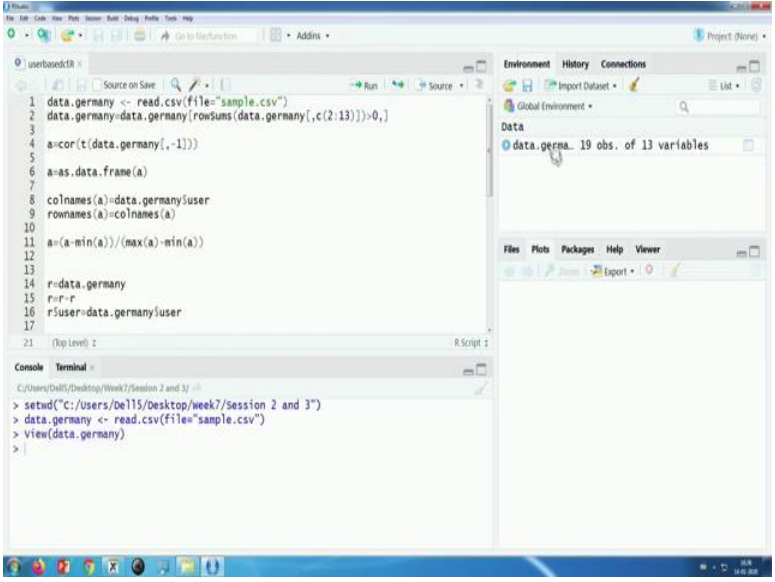
Marketing Analytics
Prof. Swagato Chatterjee
Vinod Gupta School of Management
Indian Institute of Technology, Kharagpur
Lecture 37: Recommendation Engine and Retail Analytics (Contd.)

Hello everybody, welcome to Marketing Analytics course, this is Dr. Swagato Chatterjee from VGSOM, IIT Kharagpur who is taking this course and we are in week seven and we are discussing Recommendation Engine. So, till now we have talked about item to item collaborative filtering. In this particular video, I will also talk about user to user collaborative filtering. So, how can I find out whether the two users are similar or not? And how we can deal with that?

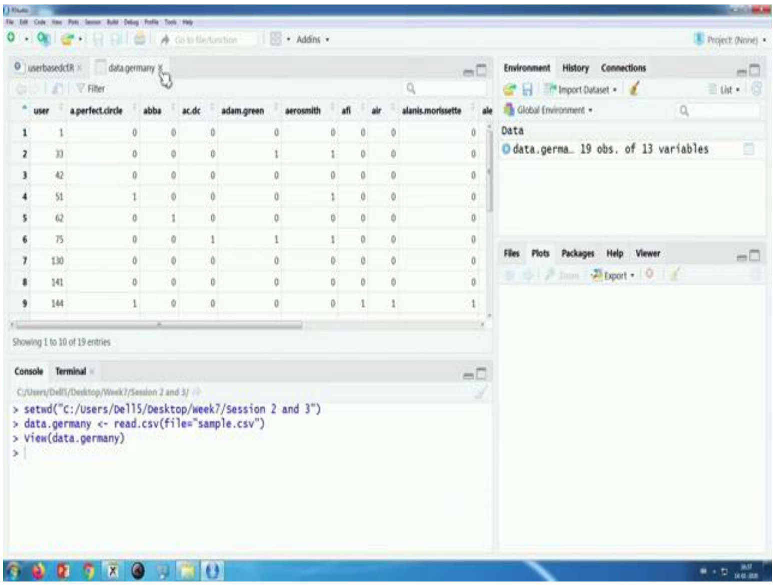
(Refer Slide Time: 0:56)



```
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user

```

```
> setwd("C:/Users/Dell5/Desktop/week7/Session 2 and 3")
> data.germany <- read.csv(file="sample.csv")
> View(data.germany)
> 
```

Data
data.germa.. 19 obs. of 13 variables



	user	a.perfect.circle	abba	ac.dc	adam.green	aerosmith	afi	air	alanis.morissette	ale
1	1	0	0	0	0	0	0	0	0	
2	33	0	0	0	1	1	0	0	0	
3	42	0	0	0	0	0	0	0	0	
4	51	1	0	0	0	1	0	0	0	
5	62	0	1	0	0	0	0	0	0	
6	75	0	0	1	1	1	0	0	0	
7	130	0	0	0	0	0	0	0	0	
8	141	0	0	0	0	0	0	0	0	
9	144	1	0	0	0	0	1	1	1	

Showing 1 to 10 of 19 entries

```
> setwd("C:/Users/Dell5/Desktop/week7/Session 2 and 3")
> data.germany <- read.csv(file="sample.csv")
> View(data.germany)
> 
```

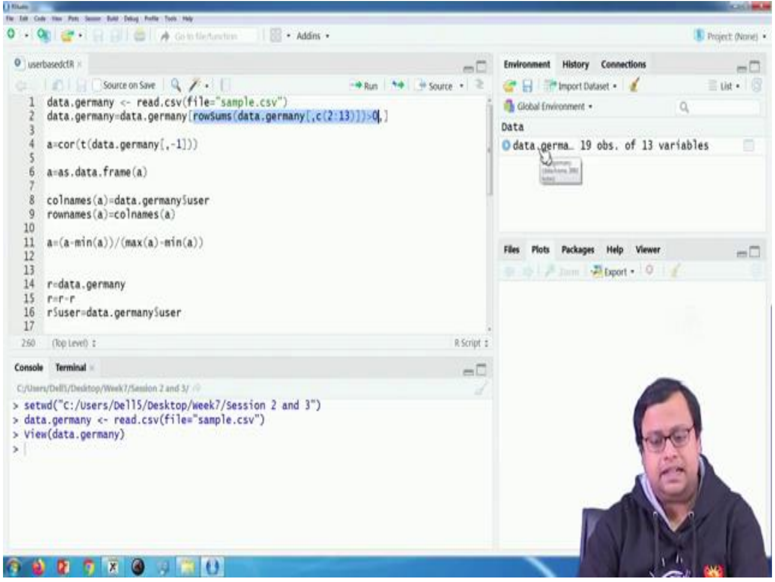
Data
data.germa.. 19 obs. of 13 variables

So, we will use the same dataset but there is something called userbasecf.R file. So, I am just opening that file using the same dataset, and we will here we will create how two users are similar to each other and based on that, we will probably try to find out that whether certain movies or in this case certain I would say songs can be recommended to a particular customer or not.

So, the dataset if you see the dataset is same, so session set, working directory, two source file location that is the first job that we want to do. And then I will read the dataset which is same sample.csv file. It is a smaller version of the bigger dataset which has 19 observations and 13 variables. So, 19 observations of various users and 13 variables are actually 13 columns. Out of them first column is the user column itself. And from then, this column till the last column, this all these columns are actually whether a particular the movie or a particular in this case a song has been consumed, has been listened by a user or not. So that is something that we are trying to do here.

So, now these 0’s and 1’s means 0 means that he has not seen the video and 1 means that he has seen the video or in this case probably the he has listened to the song and we are going ahead with that.

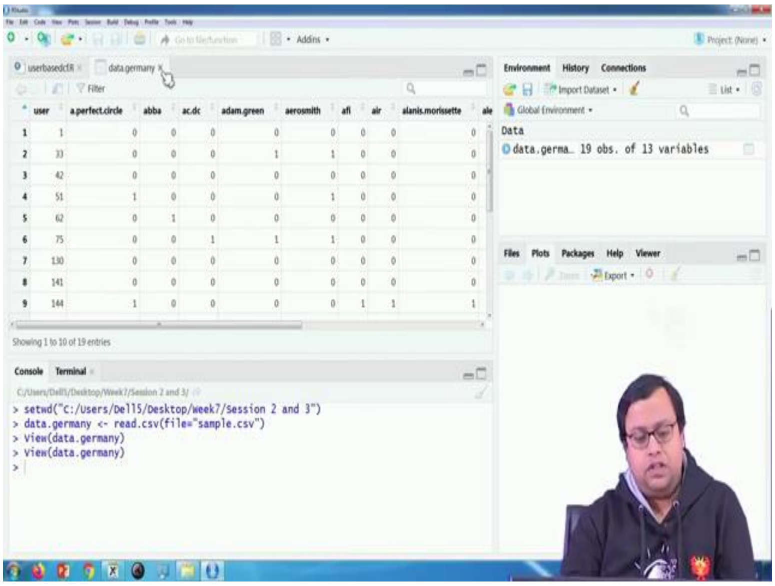
(Refer Slide Time: 2:26)



```
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user

```

```
> setwd("C:/Users/Dell5/Desktop/Week7/Session 2 and 3")
> data.germany <- read.csv(file="sample.csv")
> View(data.germany)
> 
```

data.germa.. 19 obs. of 13 variables



data.germany

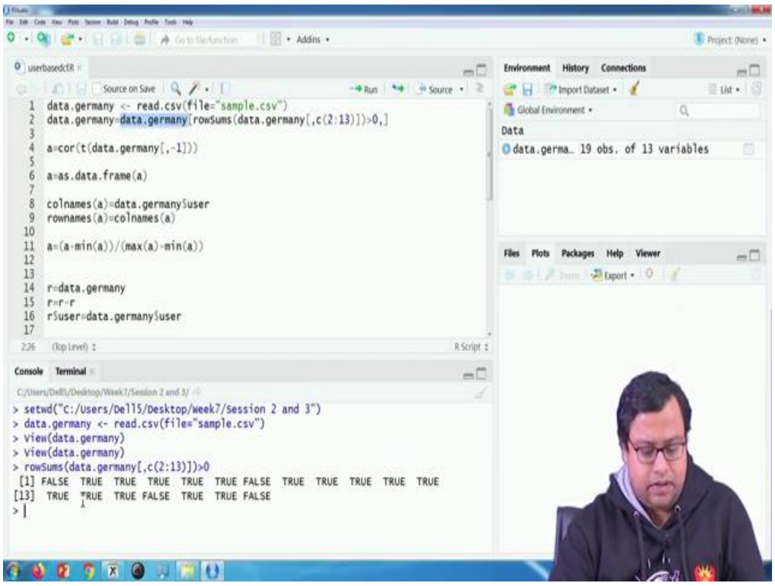
	user	a.perfect.circle	abba	ac.dc	adam.green	aerosmith	afi	air	alanis.morissette	ale
1	1	0	0	0	0	0	0	0	0	
2	33	0	0	0	1	1	0	0	0	
3	42	0	0	0	0	0	0	0	0	
4	51	1	0	0	0	1	0	0	0	
5	62	0	1	0	0	0	0	0	0	
6	75	0	0	1	1	1	0	0	0	
7	130	0	0	0	0	0	0	0	0	
8	141	0	0	0	0	0	0	0	0	
9	144	1	0	0	0	0	1	1	1	

Showing 1 to 10 of 19 entries

Data
data.germa.. 19 obs. of 13 variables

C:/Users/Dell5/Desktop/Week7/Session 2 and 3/

```
> setwd("C:/Users/Dell5/Desktop/week7/Session 2 and 3")
> data.germany <- read.csv(file="sample.csv")
> View(data.germany)
> View(data.germany)
> 
```



```
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user

```

Data
data.germa.. 19 obs. of 13 variables

C:/Users/Dell5/Desktop/Week7/Session 2 and 3/

```
> setwd("C:/Users/Dell5/Desktop/week7/Session 2 and 3")
> data.germany <- read.csv(file="sample.csv")
> View(data.germany)
> View(data.germany)
> rowSums(data.germany[,c(2:13)])>0
 [1] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
[13]  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE
> |
```

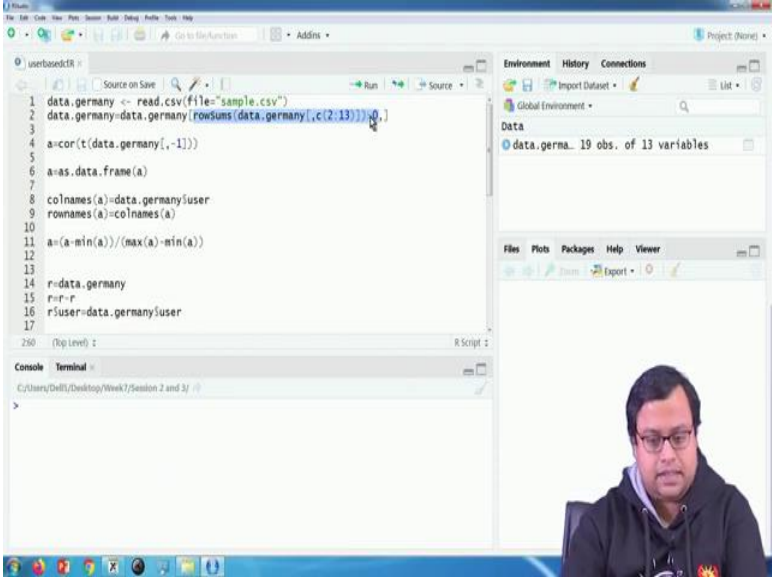
So, the first things first is we will check only for such kind of cases whether rowSums, you will see that rowSum of data Germany 2 to 13 has to be > 0. So, what is rowSum? rowSum means that by chance, if there is a dataset, where rowSum means the summation of these rows, so, row wise summation, each row, you take each row one at a time and take a summation of the values.

Now, if all the values are zero then this guy has not seen any of the movies in this particular list. If one user has not listened to or not listened any songs or listened to any movies in a particular list, then even if you get similarity matrix and blah, blah, blah, whatever you do not have any historical data for that person, and if you do not have historical data and in this context, we do not also have the demographic data of the person. So, if I have no data about that person, I cannot recommend him anything.

So, the first small job that we are doing here, which was not the case in the previous thing, we are doing column wise so, we have to check probably that column wise if the sum is zero, then that particular movie is not seen by anybody. So, similarity of that movie with any movie is zero. So, we could have dropped that movie so, that is one way. So, that is the problem of collaborative filtering as we discussed in the first video in this week. That collaborative filtering will not work when the data is new, when the user is new or the movie or whichever the product ID is new, if the product is new or the user is new, you will not get historical data of that person and user based collaborative filtering or item based collaborative filtering actually relies on your historical data. If historical data is absent for you, then you cannot use that in a collaborative filtering.

So, now here we are saying that rowSums >0 zero for whom? So, if I just copy that part, the part that I have highlighted and then paste it here and press, see there are some guys who are false. So for example, for the first row or the 13, 14, 15, 16 row or these guys so, say some people in this dataset who has not seen any movie or not listened to any music in this particular case. So that means that I have to drop them, before I go ahead with doing any kind of modeling, so that is why I am saying that if by chance they are true they only you take those row so see, I am doing yourself sub setting here, if you see that data.Germany, what did I write here? Data.Germany, this is what I have written there, data.Germany and then the third bracket, that is what I have written there, this is the third bracket.

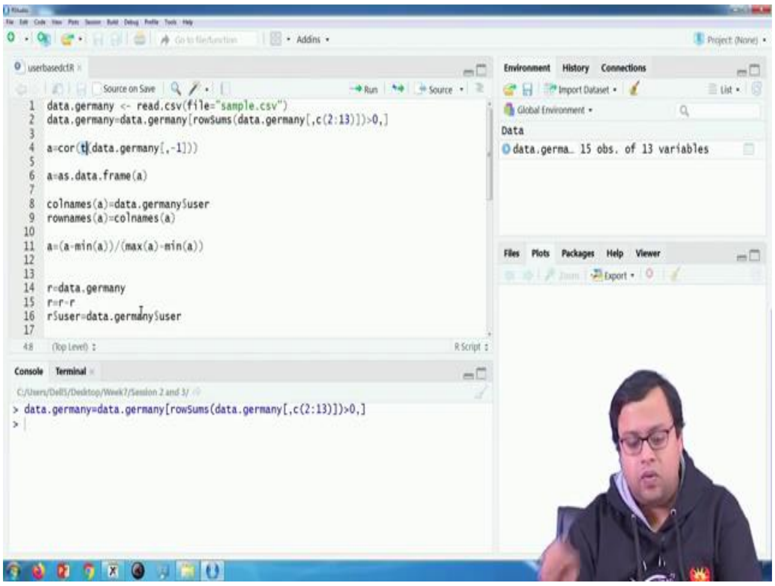
(Refer Slide Time: 5:21)



```
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user

```

C:/Users/Dell5/Desktop/Week7/Session 2 and 3/

data.germa.. 19 obs. of 13 variables



```
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user
```

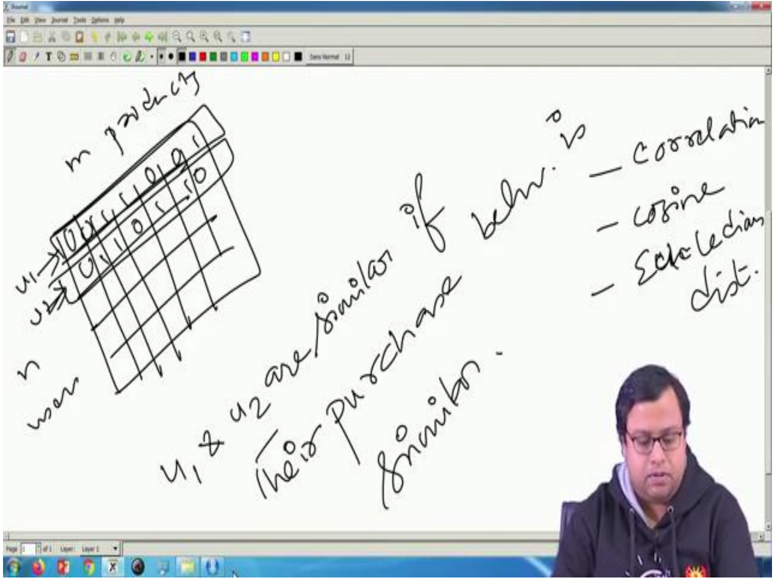
```
C:/Users/Dell5/Desktop/Week7/Session 2 and 3/
> data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]
>
```

```
data.germa.. 15 obs. of 13 variables
```

And in the third bracket means that subset of data.Germany then I have written a , , means some rows, some columns, nothing written after , that means all the columns, right and what is written before the row? In the rows, this is what it is written in the rows. So, wherever these value, the highlighted value is coming true, you are getting your result, wherever it is false, that particular guy is being dropped.

So given that if I just run this now, I got 15 observations from 19 observations it got dropped to 15 observations. So, that is fine. So, after that I can do my mathematics. So then what I am doing is I am trying to find out correlation, you can also find out cosine or whatever measurement that you want to take for, correlation is the easiest one and we understand that in marketing, but in the real world, cosine is the most prominently used matrix for finding out the similarity between two groups or in this case two vectors. So correlation of T of germany transpose, why I have taken transpose? Because now I am taking row wise.

(Refer Slide Time: 6:40)

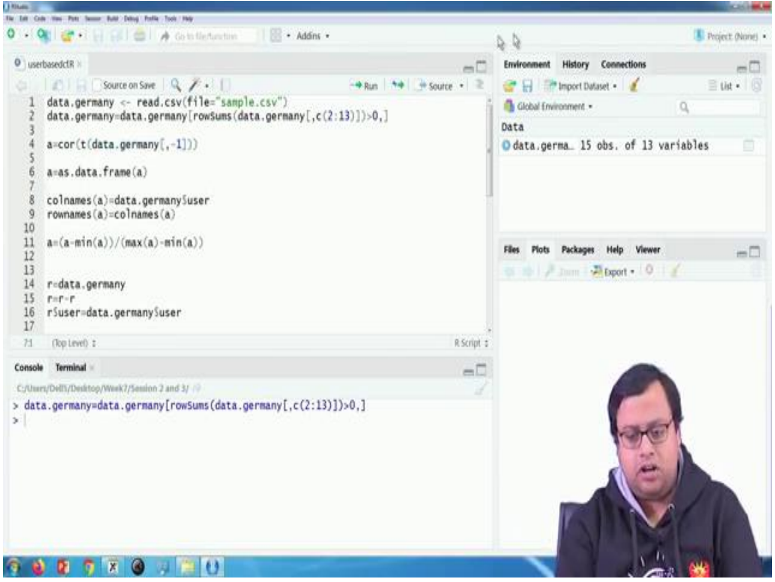


u1 & u2 are similar if their purchase behv. is similar.

So, if you remember, if I have, this is my data set. And if I have n number of users and m number of products, so these are 01, 01, 01 rows, earlier I was doing column wise, there I was doing like this column wise. Now, it is 0 1 row, so 0, 0, 1, 1, 0, 0, 1 then 0, 1, 1, 0, 1, 1, 0 or so on something like that, so, this guy and this guy the user one and user two are similar, if user one and user two are similar, if their purchase behavior is similar, now you can measure, check this is 0 1 column, right, row means vector, 0 1 vector.

So the similarity can be measured by various ways, it can be measured by correlation, it can be measured by cosine, it can be also measured by let us say Standardized Euclidean distance, Euclidean distance. So, all this is applicable so, which ever you want to do, you can do it. So, here what I am doing is I am doing coordination.

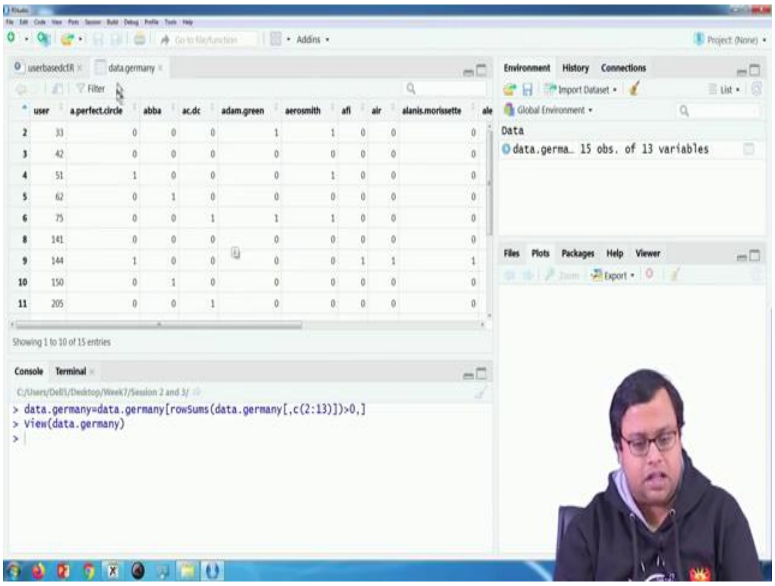
(Refer Slide Time: 8:07)



```r
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user

```

Data
data.germa.. 15 obs. of 13 variables

```
C:/Users/Dell5/Desktop/Week7/Session 2 and 3/
> data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]
> 
```

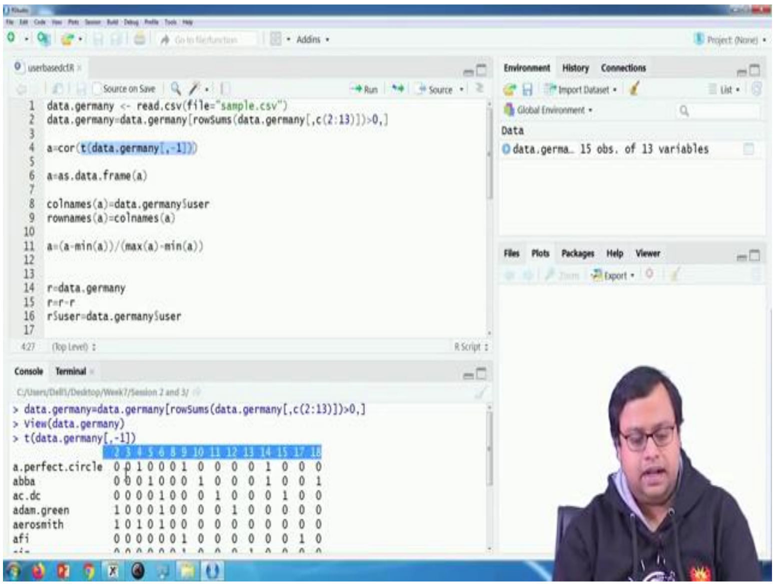


	user	a.perfect.circle	abba	ac.dc	adam.green	aerosmith	afi	air	alanis.morissette	ale
2	33	0	0	0	1	1	0	0	0	
3	42	0	0	0	0	0	0	0	0	
4	51	1	0	0	0	1	0	0	0	
5	62	0	1	0	0	0	0	0	0	
6	75	0	0	1	1	1	0	0	0	
8	141	0	0	0	0	0	0	0	0	
9	144	1	0	0	0	0	1	1	1	
10	150	0	1	0	0	0	0	0	0	
11	205	0	0	1	0	0	0	0	0	

Showing 1 to 10 of 15 entries

Data
data.germa.. 15 obs. of 13 variables

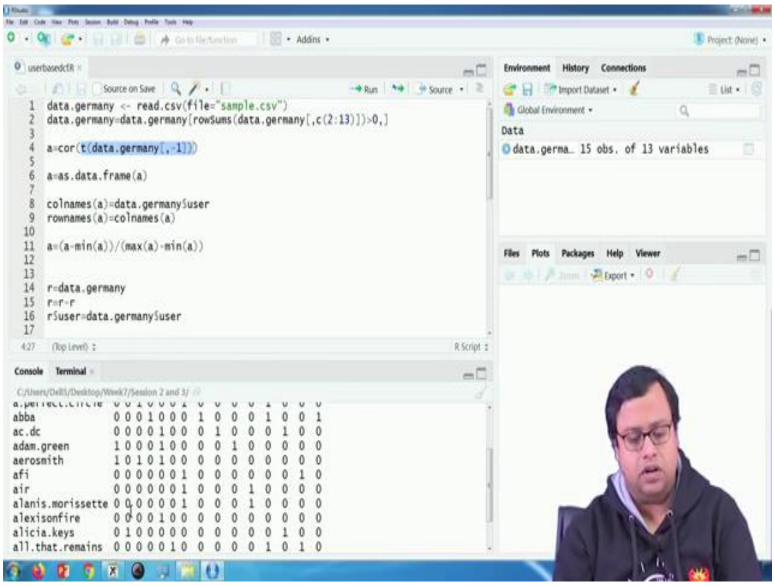
```
C:/Users/Dell5/Desktop/Week7/Session 2 and 3/
> data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]
> View(data.germany)
> 
```



```r
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user

```

Data
data.germa.. 15 obs. of 13 variables

```
C:/Users/Dell5/Desktop/Week7/Session 2 and 3/
> data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]
> View(data.germany)
> t(data.germany[,-1])
                2 3 4 5 6 8 9 10 11 12 13 14 15 17 18
a.perfect.circle 0 0 1 0 0 0 1  0  0  0  0  1  0  0  0
abba             0 0 0 1 0 0 0  1  0  0  0  1  0  0  1
ac.dc            0 0 0 0 1 0 0  0  1  0  0  0  1  0  0
adam.green       1 0 0 0 1 0 0  0  0  1  0  0  0  0  0
aerosmith        1 0 1 0 1 0 0  0  0  0  0  0  0  0  0
afi              0 0 0 0 0 0 1  0  0  0  0  0  0  1  0
```



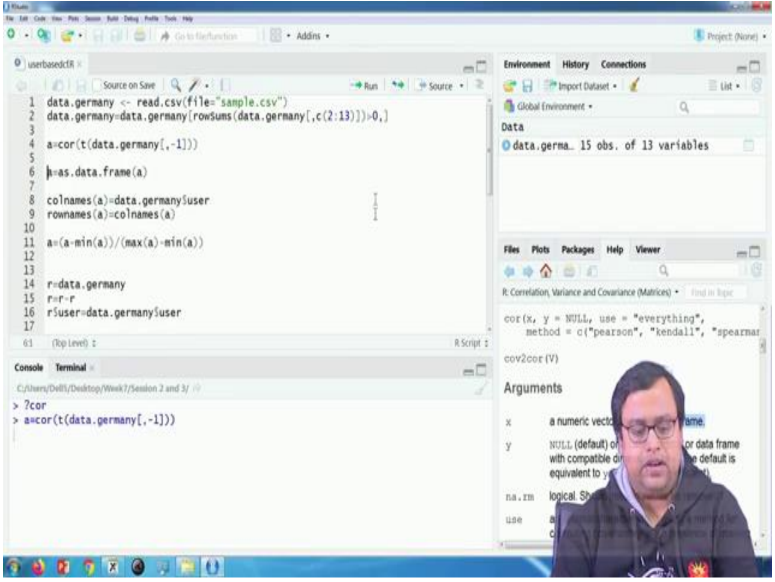
```
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user
```

Data
data.germa.. 15 obs. of 13 variables

```
abba              0 0 0 1 0 0 0  1  0  0  0  1  0  0  1
ac.dc             0 0 0 0 1 0 0  0  1  0  0  0  1  0  0
adam.green        1 0 0 0 1 0 0  0  0  1  0  0  0  0  0
aerosmith         1 0 1 0 1 0 0  0  0  0  0  0  0  0  0
afi               0 0 0 0 0 0 1  0  0  0  0  0  0  1  0
air               0 0 0 0 0 0 1  0  0  0  1  0  0  0  0
alanis.morissette 0 0 0 0 0 0 1  0  0  0  1  0  0  0  0
alexisonfire      0 0 0 0 1 0 0  0  0  0  0  0  0  0  0
alicia.keys       0 1 0 0 0 0 0  0  0  0  0  0  1  0  0
all.that.remains  0 0 0 0 0 1 0  0  0  0  0  1  0  1  0
```

So, I am doing transposition, data.germany[,-1], means first column you drop, first column gets dropped, that means this particular column gets dropped, then whatever this guy had, I am taking a transpose of that. So, that looks like, that looks like if I just copy this paste it here, that looks like this. So, the column, the movie names came here, and this is the customer guys. And these are his purchase behavior, his purchase behavior, third guy's purchase behavior and so on. So, for this and this if I take 0, 0, 1, 1, 0, 0, 0, 1, 1, 0 and so on, and then 0, 0, 0, 0, 0, 0 and so on for the third person that means the second guy in this case.

(Refer Slide Time: 9:15)



```
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user
```

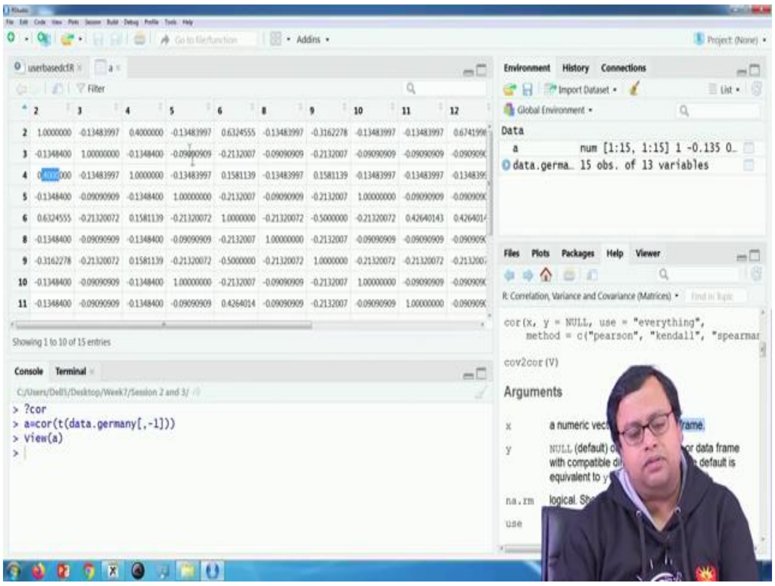
```
> ?cor
> a=cor(t(data.germany[,-1]))
```

Data
data.germa.. 15 obs. of 13 variables

R: Correlation, Variance and Covariance (Matrices)

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearmar
cov2cor(V)
```

Arguments



	2	3	4	5	6	8	9	10	11	12
2	1.0000000	-0.13483997	0.4000000	-0.13483997	0.6324555	-0.13483997	-0.3162278	-0.13483997	-0.13483997	0.674199
3	-0.1348400	1.00000000	-0.1348400	-0.09090909	-0.2132007	-0.09090909	-0.2132007	-0.09090909	-0.09090909	-0.090909
4	0.4000000	-0.13483997	1.0000000	-0.13483997	0.1581139	-0.13483997	0.1581139	-0.13483997	-0.13483997	-0.134839
5	-0.1348400	-0.09090909	-0.1348400	1.00000000	-0.2132007	-0.09090909	-0.2132007	1.00000000	-0.09090909	-0.090909
6	0.6324555	-0.21320072	0.1581139	-0.21320072	1.0000000	-0.21320072	-0.5000000	-0.21320072	0.42640143	0.426401
8	-0.1348400	-0.09090909	-0.1348400	-0.09090909	-0.2132007	1.00000000	-0.2132007	-0.09090909	-0.09090909	-0.090909
9	-0.3162278	-0.21320072	0.1581139	-0.21320072	-0.5000000	-0.21320072	1.0000000	-0.21320072	-0.21320072	-0.213200
10	-0.1348400	-0.09090909	-0.1348400	1.00000000	-0.2132007	-0.09090909	-0.2132007	1.00000000	-0.09090909	-0.090909
11	-0.1348400	-0.09090909	-0.1348400	-0.09090909	0.4264014	-0.09090909	-0.2132007	-0.09090909	1.00000000	-0.090909

Showing 1 to 10 of 15 entries

C:/Users/Dell5/Desktop/Week7/Session 2 and 3/

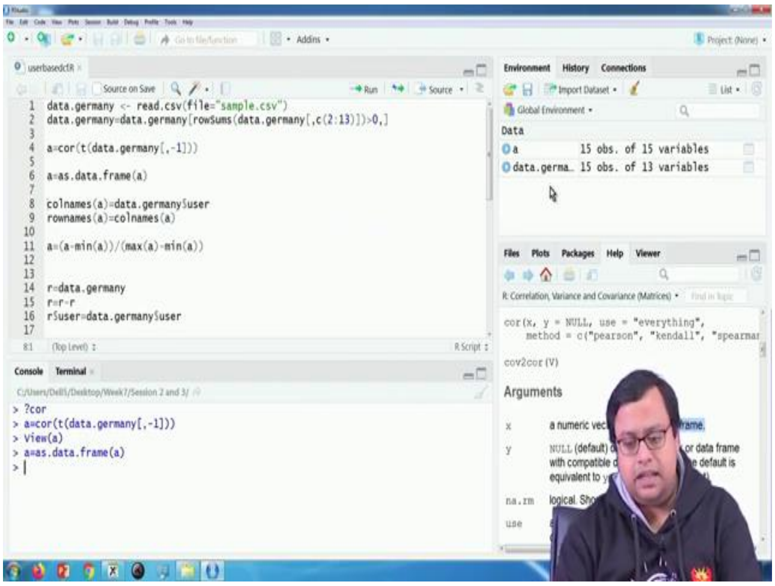
```
> ?cor
> a=cor(t(data.germany[,-1]))
> View(a)
>
```

Data

```
a          num [1:15, 1:15] 1 -0.135 0..
data.germa.. 15 obs. of 13 variables
```

R: Correlation, Variance and Covariance (Matrices)

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearmar
cov2cor(V)
```



```
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user

```

C:/Users/Dell5/Desktop/Week7/Session 2 and 3/

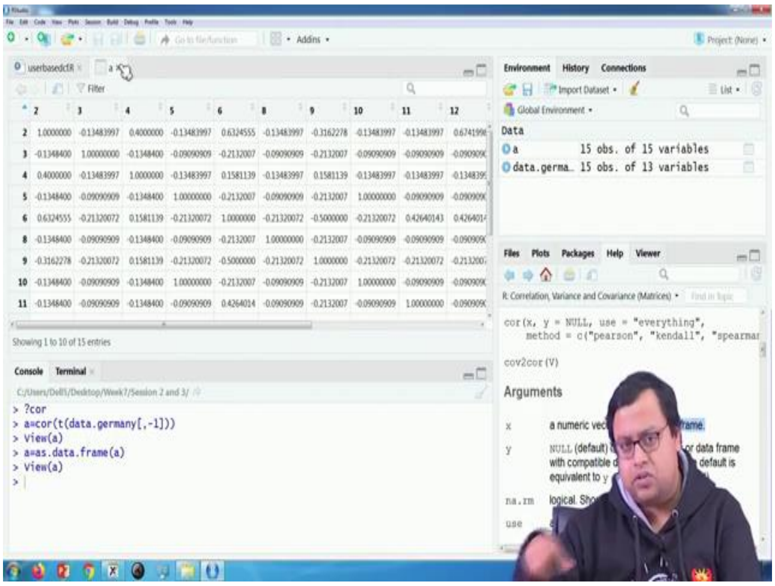
```
> ?cor
> a=cor(t(data.germany[,-1]))
> View(a)
> a=as.data.frame(a)
>
```

Data

```
a          15 obs. of 15 variables
data.germa.. 15 obs. of 13 variables
```

R: Correlation, Variance and Covariance (Matrices)

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearmar
cov2cor(V)
```



	2	3	4	5	6	8	9	10	11	12
2	1.0000000	-0.13483997	0.4000000	-0.13483997	0.6324555	-0.13483997	-0.3162278	-0.13483997	-0.13483997	0.674199
3	-0.1348400	1.00000000	-0.1348400	-0.09090909	-0.2132007	-0.09090909	-0.2132007	-0.09090909	-0.09090909	-0.090909
4	0.4000000	-0.13483997	1.0000000	-0.13483997	0.1581139	-0.13483997	0.1581139	-0.13483997	-0.13483997	-0.134839
5	-0.1348400	-0.09090909	-0.1348400	1.00000000	-0.2132007	-0.09090909	-0.2132007	1.00000000	-0.09090909	-0.090909
6	0.6324555	-0.21320072	0.1581139	-0.21320072	1.0000000	-0.21320072	-0.5000000	-0.21320072	0.42640143	0.426401
8	-0.1348400	-0.09090909	-0.1348400	-0.09090909	-0.2132007	1.00000000	-0.2132007	-0.09090909	-0.09090909	-0.090909
9	-0.3162278	-0.21320072	0.1581139	-0.21320072	-0.5000000	-0.21320072	1.0000000	-0.21320072	-0.21320072	-0.213200
10	-0.1348400	-0.09090909	-0.1348400	1.00000000	-0.2132007	-0.09090909	-0.2132007	1.00000000	-0.09090909	-0.090909
11	-0.1348400	-0.09090909	-0.1348400	-0.09090909	0.4264014	-0.09090909	-0.2132007	-0.09090909	1.00000000	-0.090909

Showing 1 to 10 of 15 entries

C:/Users/Dell5/Desktop/Week7/Session 2 and 3/

```
> ?cor
> a=cor(t(data.germany[,-1]))
> View(a)
> a=as.data.frame(a)
> View(a)
>
```

Data

```
a          15 obs. of 15 variables
data.germa.. 15 obs. of 13 variables
```

R: Correlation, Variance and Covariance (Matrices)

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearmar
cov2cor(V)
```

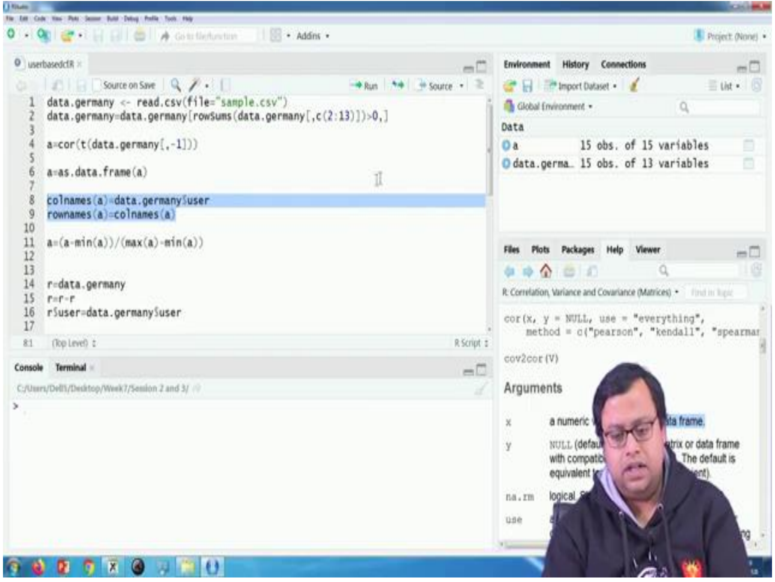
If I take a correlation between that, that will give me the similarity between these people. So, if I just try to find out correlation using the function called COR C, O, R which is available in the base package, so, COR will be the correlation function, the syntax is COR and it is saying that okay, if you give the whole dataset in COR, you can give x whichever, x is what? x is a numeric vector, matrix or data frame. If it is a numeric matrix or data frame, it will find out each column and take the correlation of those columns. If it is a vector, then you have to give another vector. So, you can do that and I am running this and saving the result in a so, if I run this a looks like this.

So, this is basically the values that you are seeing here or basically correlation values with 2 and 2 the correlation between 2 and 2 is 1, 3 and 3 is 1, 4 and 4 is 1, 5 and 5 is 1 and so on. And all the off-diagonal elements are basically mirror image over the diagonal that means, if this is 1.13487, this is also 1.81- 0.183483997 actually this is also 3997 and if it is 0.4 this is also 0.4 and so on. So, those kind of values you will get.

Now, I am converting this a to a data frame because that helps me in doing further some, some amount of analysis and some extra advantage data frame has over matrices, we have discussed about that in a different class. So, that is something that we are doing, we are changing it to a data frame. Now, after changing it to a data frame, what I am doing is, I am actually putting the column names also. So, see here the user names are not coming properly, it is two, three, these are not exactly the user IDs, these are actually the serial numbers of that particular column.

So second column, third column, first column got dropped, then second column, third column etcetera that 2, 3, 4 has come.

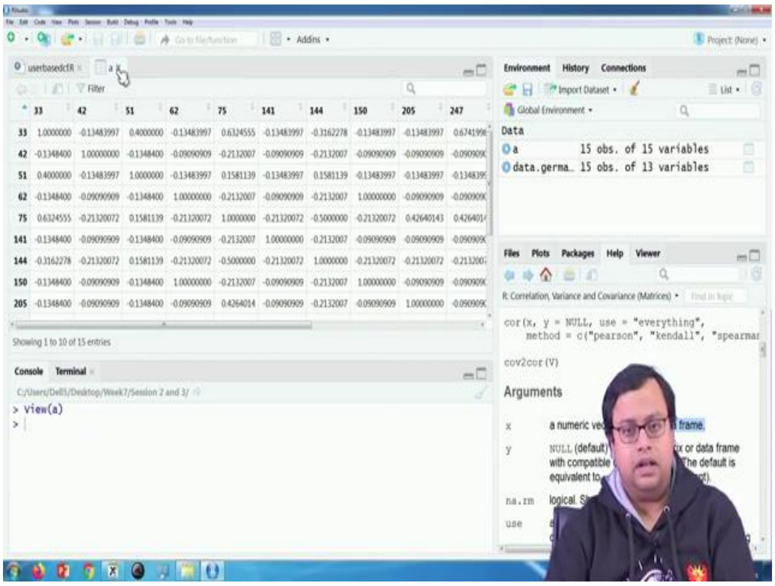
(Refer Slide Time: 11:22)



```
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user
```

```
a        15 obs. of 15 variables
data.germa.. 15 obs. of 13 variables
```

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearman
```



	33	42	51	62	75	141	144	150	205	247
33	1.0000000	-0.13483997	0.4000000	-0.13483997	0.6324555	-0.13483997	-0.3162278	-0.13483997	-0.13483997	0.674199
42	-0.1348400	1.00000000	-0.1348400	-0.09090909	-0.2132007	-0.09090909	-0.2132007	-0.09090909	-0.09090909	-0.09090
51	0.4000000	-0.13483997	1.0000000	-0.13483997	0.1581139	-0.13483997	0.1581139	-0.13483997	-0.13483997	-0.134839
62	-0.1348400	-0.09090909	-0.1348400	1.00000000	-0.2132007	-0.09090909	-0.2132007	1.00000000	-0.09090909	-0.09090
75	0.6324555	-0.21320072	0.1581139	-0.21320072	1.0000000	-0.21320072	-0.5000000	-0.21320072	0.42640143	0.426401
141	-0.1348400	-0.09090909	-0.1348400	-0.09090909	-0.2132007	1.00000000	-0.2132007	-0.09090909	-0.09090909	-0.09090
144	-0.3162278	-0.21320072	0.1581139	-0.21320072	-0.5000000	-0.21320072	1.0000000	-0.21320072	-0.21320072	-0.21320
150	-0.1348400	-0.09090909	-0.1348400	1.00000000	-0.2132007	-0.09090909	-0.2132007	1.00000000	-0.09090909	-0.09090
205	-0.1348400	-0.09090909	-0.1348400	-0.09090909	0.4264014	-0.09090909	-0.2132007	-0.09090909	1.00000000	-0.09090

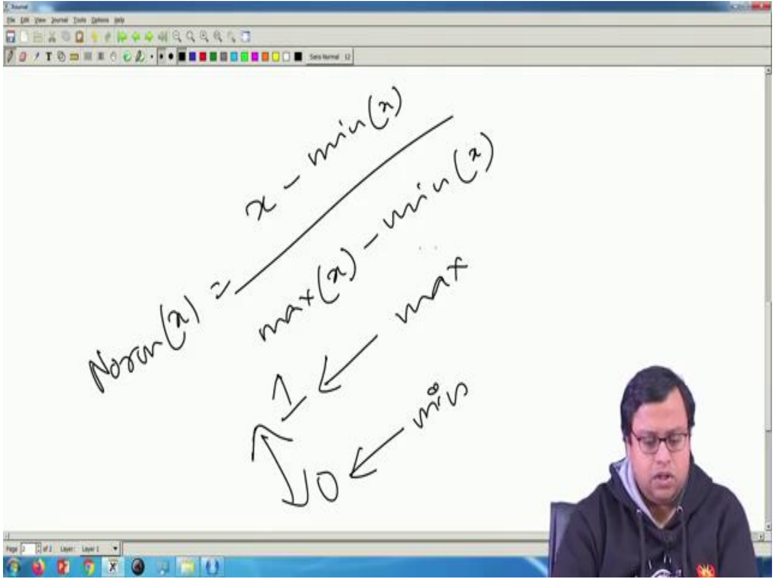
Showing 1 to 10 of 15 entries

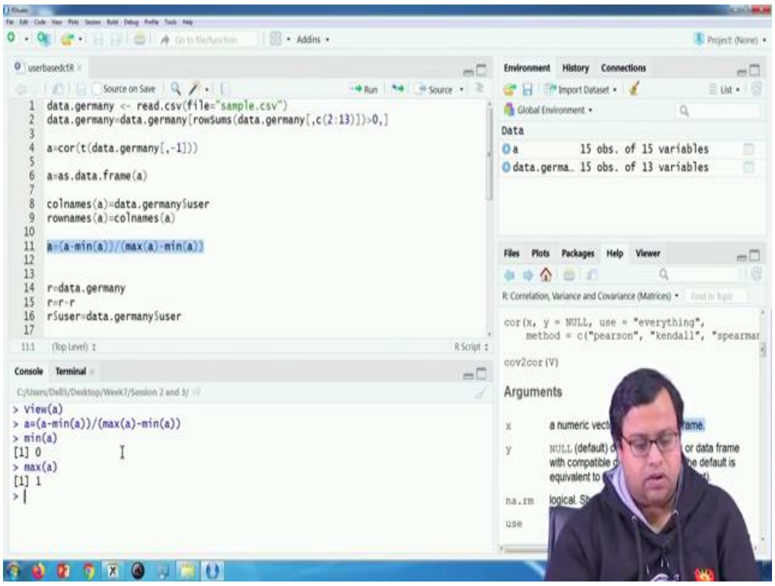
```
> View(a)
>
```

So, I am saying columns name of a is datadollar. Germany dollar user, data.Germany dollar user is actual user IDs of those people, user id number 33, 42, 51, 62 and etcetera. So, that should come into the column name and also in the row name. So, if I run these two lines, first, the column names gets changed, then when I write row names =column names then the row names also get changed.

So, that is what we will get here. So, now you see it is 33, 42, 53, 62 and so on and here also 33, 42, 53, 62 and so on so, that is what I am getting here, in this particular case. So whatever it is just to make sure that I can visualize it properly, I can see it properly. And I can understand that. So, what did I get till now? I got, instead of previous one while we are doing item to item collaborative filtering, and I was getting a similarity matrix for one item over another item, here I am getting similarity matrix for the user, one user towards other users. And then I will have to find out who are the top five users.

(Refer Slide Time: 12:45)





```
data.germany <- read.csv(file="sample.csv")
data.germany=data.germany[rowSums(data.germany[,c(2:13)])>0,]

a=cor(t(data.germany[,-1]))

a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user

```

```
> View(a)
> a=(a-min(a))/(max(a)-min(a))
> min(a)
[1] 0
> max(a)
[1] 1
> |
```

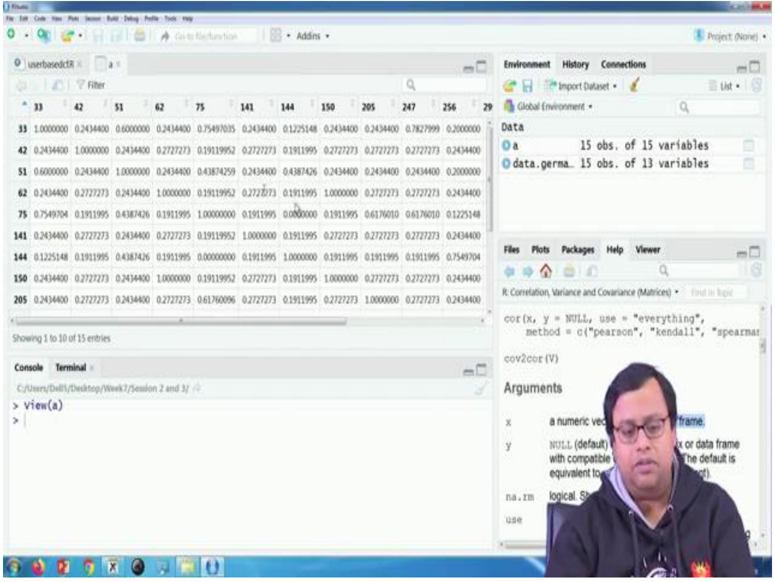
```
Data
a              15 obs. of 15 variables
data.germa..   15 obs. of 13 variables
```

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearmar
```

So I am maximizing, I am putting a normalization, if you remember, the normalization that we did before is normalized x Norm(x)=x-min(x)/max(x)-min(x) what does it do? It actually binds the normalized x × 0 and 1, so the minimum value comes to be 0 and the maximum value comes to be 1. So, that is what is also happening here, this is something that we are trying to do. So, if you check here, I am doing is a = -min of a and now it is a matrix. So, it will find out the minimum of a in the whole matrix by max(a)-min(a).

So, the maximum entry in this currently in this is a 0 and minimum entry is 0 and maximum entry is 1 and that is how it has changed the whole a column.

(Refer Slide Time: 13:40)



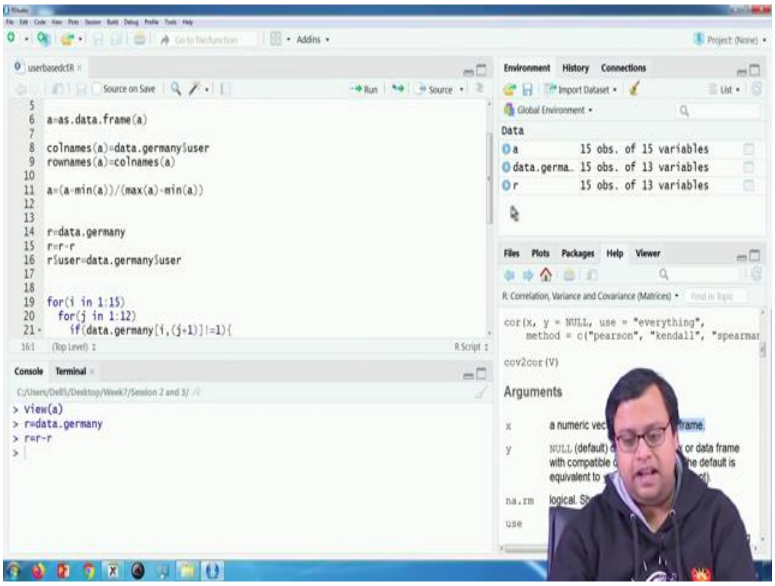
	33	42	51	62	75	141	144	150	205	247	256
33	1.0000000	0.2434400	0.6000000	0.2434400	0.75497035	0.2434400	0.1225148	0.2434400	0.2434400	0.7827999	0.2000000
42	0.2434400	1.0000000	0.2434400	0.2727273	0.19119952	0.2727273	0.1911995	0.2727273	0.2727273	0.2727273	0.2434400
51	0.6000000	0.2434400	1.0000000	0.2434400	0.43874259	0.2434400	0.4387426	0.2434400	0.2434400	0.2434400	0.2000000
62	0.2434400	0.2727273	0.2434400	1.0000000	0.19119952	0.2727273	0.1911995	1.0000000	0.2727273	0.2727273	0.2434400
75	0.7549704	0.1911995	0.4387426	0.1911995	1.00000000	0.1911995	0.0000000	0.1911995	0.6176010	0.6176010	0.1225148
141	0.2434400	0.2727273	0.2434400	0.2727273	0.19119952	1.0000000	0.1911995	0.2727273	0.2727273	0.2727273	0.2434400
144	0.1225148	0.1911995	0.4387426	0.1911995	0.00000000	0.1911995	1.0000000	0.1911995	0.1911995	0.1911995	0.7549704
150	0.2434400	0.2727273	0.2434400	1.0000000	0.19119952	0.2727273	0.1911995	1.0000000	0.2727273	0.2727273	0.2434400
205	0.2434400	0.2727273	0.2434400	0.2727273	0.61760096	0.2727273	0.1911995	0.2727273	1.0000000	0.2727273	0.2434400

Showing 1 to 10 of 15 entries

```
> View(a)
>
```

```
a          15 obs. of 15 variables
data.germa..  15 obs. of 13 variables
```

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearmar
```



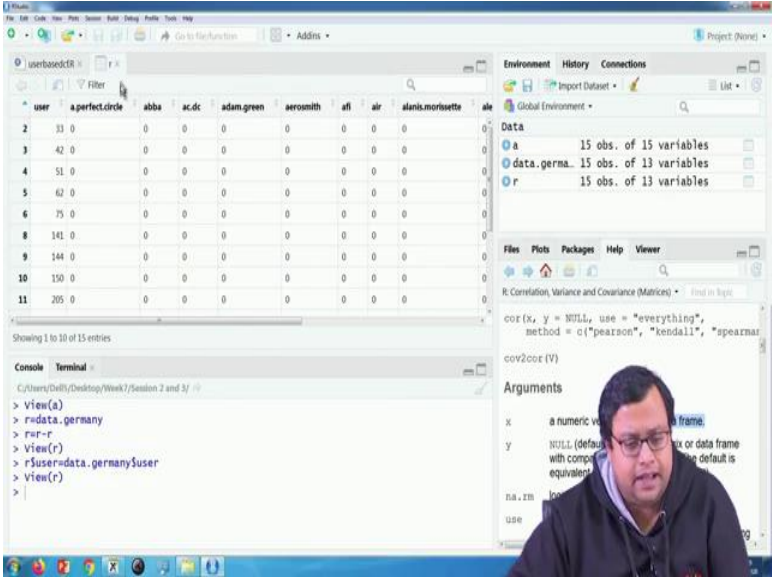
```
a=as.data.frame(a)

colnames(a)=data.germany$user
rownames(a)=colnames(a)

a=(a-min(a))/(max(a)-min(a))


r=data.germany
r=r-r
r$user=data.germany$user


for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){
```

```
> View(a)
> r=data.germany
> r=r-r
>
```

```
a          15 obs. of 15 variables
data.germa..  15 obs. of 13 variables
r          15 obs. of 13 variables
```

So, if you see this, you will see that all the values have changed, there is no negative value anymore. So, no negative value means that we have to track that in a similar way. Fair enough. So now, the next step is to create the I would say the movies importance value or something like that. So, I am creating a recommendation matrix called r, r=data.germany why? Because then the dimensions remain same. And then r=r-1 means whatever entry there was in r you actually remove them, all of them and then r$user=data.germany$user that means basically, it looks like this:

(Refer Slide Time: 14:24)



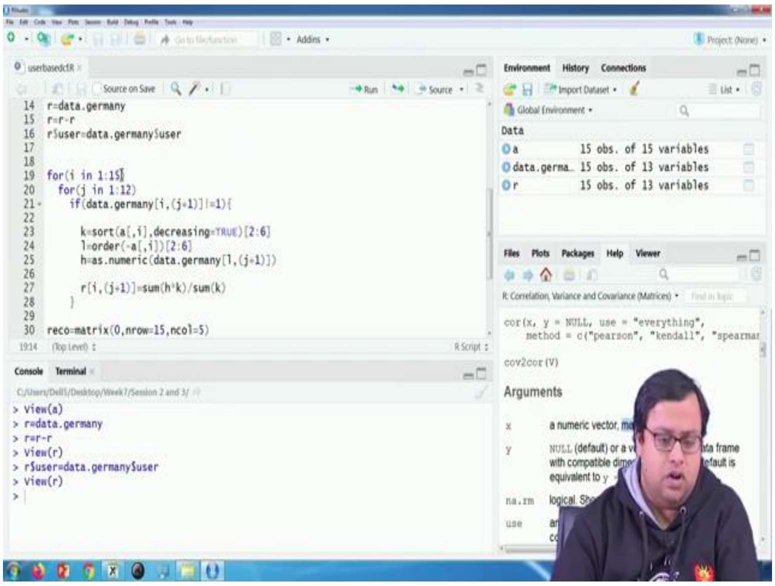
	user	a.perfect.circle	abba	ac.dc	adam.green	aerosmith	afi	air	alanis.morissette
2	33	0	0	0	0	0	0	0	0
3	42	0	0	0	0	0	0	0	0
4	51	0	0	0	0	0	0	0	0
5	62	0	0	0	0	0	0	0	0
6	75	0	0	0	0	0	0	0	0
8	141	0	0	0	0	0	0	0	0
9	144	0	0	0	0	0	0	0	0
10	150	0	0	0	0	0	0	0	0
11	205	0	0	0	0	0	0	0	0

Showing 1 to 10 of 15 entries

```
> View(a)
> r=data.germany
> r=r-r
> View(r)
> r$user=data.germany$user
> View(r)
>
```

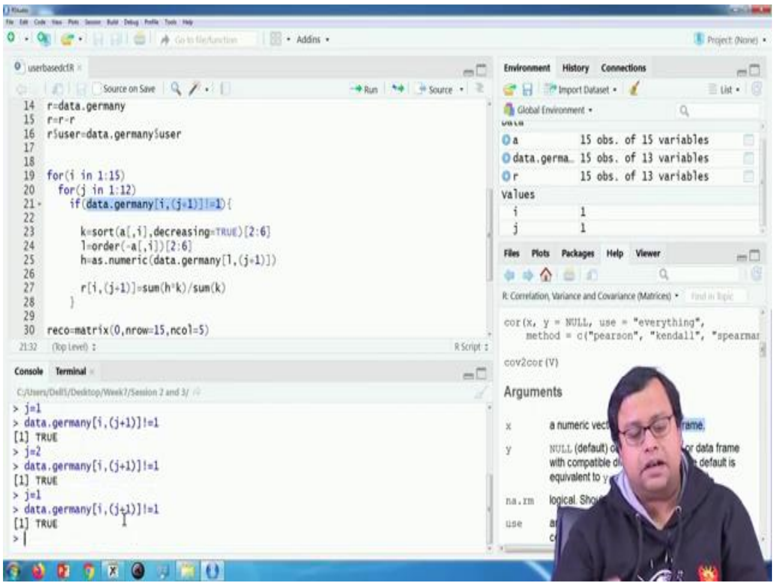
```
Data
a            15 obs. of 15 variables
data.germa.. 15 obs. of 13 variables
r            15 obs. of 13 variables
```

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearmar
cov2cor(V)
```



```
r=data.germany
r=r-r
r$user=data.germany$user


for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){

      k=sort(a[,i],decreasing=TRUE)[2:6]
      l=order(-a[,i])[2:6]
      h=as.numeric(data.germany[l,(j+1)])

      r[i,(j+1)]=sum(h*k)/sum(k)
    }

reco=matrix(0,nrow=15,ncol=5)
```

```
> View(a)
> r=data.germany
> r=r-r
> View(r)
> r$user=data.germany$user
> View(r)
>
```



```
r=data.germany
r=r-r
r$user=data.germany$user


for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){

      k=sort(a[,i],decreasing=TRUE)[2:6]
      l=order(-a[,i])[2:6]
      h=as.numeric(data.germany[l,(j+1)])

      r[i,(j+1)]=sum(h*k)/sum(k)
    }

reco=matrix(0,nrow=15,ncol=5)
```

```
> j=1
> data.germany[i,(j+1)]!=1
[1] TRUE
> j=2
> data.germany[i,(j+1)]!=1
[1] TRUE
> j=1
> data.germany[i,(j+1)]!=1
[1] TRUE
>
```

```
Values
i    1
j    1
```

So, all these values are zero, the column names are similar to data . Germany, the first column is absolutely same to data . Germany which is user. Now, what I will do is, I will find out how much is the, for each user let us say user number 33, how much is his probability or propensity or attractiveness towards a perfect circle? Then how much is for a b b a, how much is for ac . dc and so on, I will populate all those and then I will sort them up each row at a time.

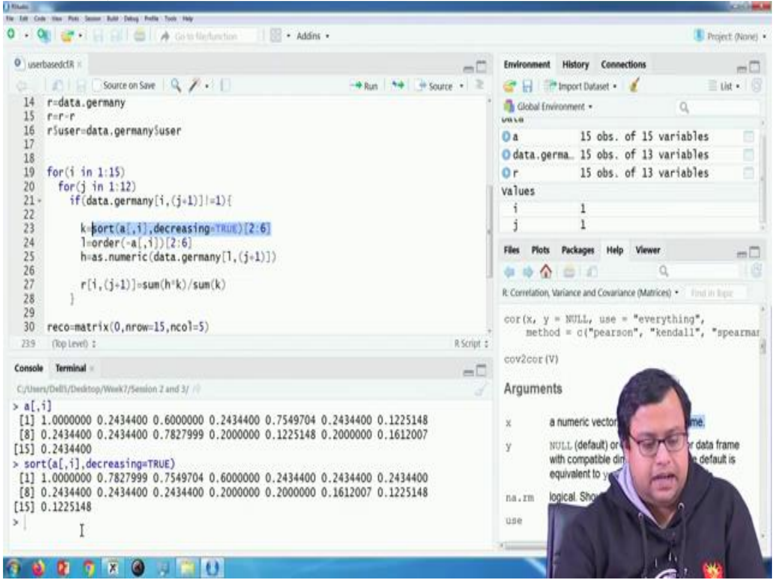
So, I will first populate so, if you think that, okay, 33 should see 10 movies, these are the various 10 movies at various levels of attractiveness, these movies are something that user number 33 should see, then those are the movies where some values will come in this particular matrix. And then when I pick up the only the row of 33 and then sort from highest to lowest, I get which movie has the highest one, which movie is the lowest one, this part is similar to the previous one but remember there were small tweaks in the presentation.

So, what I am doing is carefully you see, for i=1:15, why 1 to 15? Because there are 15 users at this moment, and for j =j=1:12, why 12? Because there are 12 movies at this moment and in this r also you will see that there, if I remove the first column, this is actually 15 × 12 matrix. So, I will populate that so, for i = 1 : 15 and for j = 1 : 12, if the same thing if(data.germany[I,(j+1)], his historical purchase data. So, whether ith customer has seen jth movie, why j + 1? Because first column in data . Germany is the user column.

So, j + 1th column is actually talking about the jth movie. So, i , j + 1 that means ith row, j + 1th column, what is the value there in data . Germany? If that value is 1, that means ith person has already seen j + 1th movie, then I will not recommend him, then corresponding value will remain 0 in my r matrix or in this call r data frame, it will not change, exactly what happened in the previous video, but by chance if if(data.germany[I,(j+1)])!=1, then the rest of the part comes into the picture, what do I do?

I find out that what is the historical purchase data of this person, of ith person and then I see that okay, so carefully see, so we do not do that, why? We see that what is the purchase data of a, means the person who is closest to the ith person, so carefully see, let us say i=1,j=1, I am just assuming so, then data . Germany, if it is 1 of 0, I will say so this is true. So, then that will not work. So let us j = 2 then != 1, okay, it is != 1, sorry. So, j = 1 is what we will try, sorry. So, this is what? So, it is true that means it comes into the loop, i = 1, j = 1 currently.

(Refer Slide Time: 18:27)

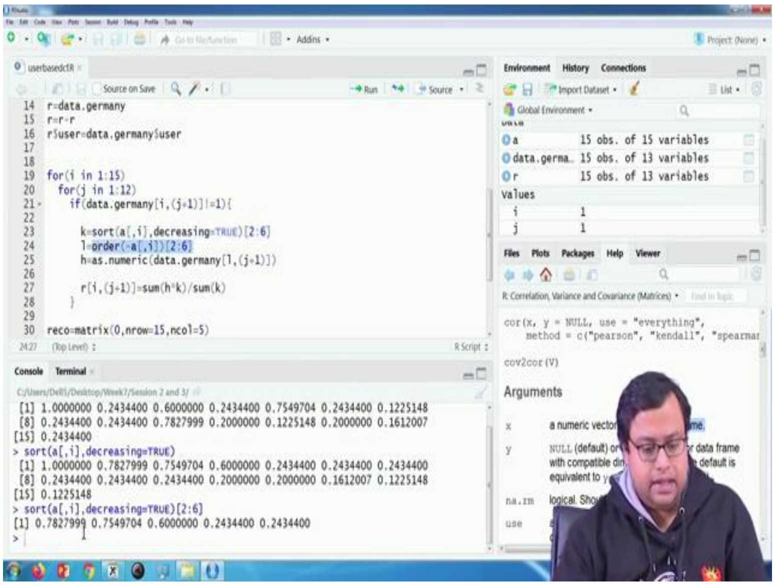


```
r=data.germany
r=r-r
r$user=data.germany$user


for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){

      k=sort(a[,i],decreasing=TRUE)[2:6]
      l=order(-a[,i])[2:6]
      h=as.numeric(data.germany[l,(j+1)])

      r[i,(j+1)]=sum(h*k)/sum(k)
    }

reco=matrix(0,nrow=15,ncol=5)
```

```
> a[,i]
 [1] 1.0000000 0.2434400 0.6000000 0.2434400 0.7549704 0.2434400 0.1225148
 [8] 0.2434400 0.2434400 0.7827999 0.2000000 0.1225148 0.2000000 0.1612007
[15] 0.2434400
> sort(a[,i],decreasing=TRUE)
 [1] 1.0000000 0.7827999 0.7549704 0.6000000 0.2434400 0.2434400 0.2434400
 [8] 0.2434400 0.2434400 0.2434400 0.2000000 0.2000000 0.1612007 0.1225148
[15] 0.1225148
> 
```

```
a            15 obs. of 15 variables
data.germa.. 15 obs. of 13 variables
r            15 obs. of 13 variables
Values
i            1
j            1
```

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearman
cov2cor(V)
```

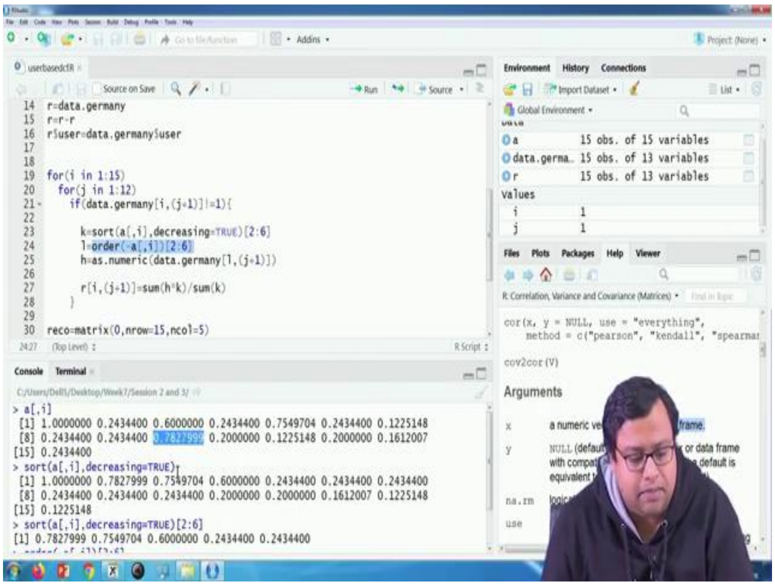


```
r=data.germany
r=r-r
r$user=data.germany$user


for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){

      k=sort(a[,i],decreasing=TRUE)[2:6]
      l=order(-a[,i])[2:6]
      h=as.numeric(data.germany[l,(j+1)])

      r[i,(j+1)]=sum(h*k)/sum(k)
    }

reco=matrix(0,nrow=15,ncol=5)
```

```
 [1] 1.0000000 0.2434400 0.6000000 0.2434400 0.7549704 0.2434400 0.1225148
 [8] 0.2434400 0.2434400 0.7827999 0.2000000 0.1225148 0.2000000 0.1612007
[15] 0.2434400
> sort(a[,i],decreasing=TRUE)
 [1] 1.0000000 0.7827999 0.7549704 0.6000000 0.2434400 0.2434400 0.2434400
 [8] 0.2434400 0.2434400 0.2434400 0.2000000 0.2000000 0.1612007 0.1225148
[15] 0.1225148
> sort(a[,i],decreasing=TRUE)[2:6]
[1] 0.7827999 0.7549704 0.6000000 0.2434400 0.2434400
> 
```

```
a            15 obs. of 15 variables
data.germa.. 15 obs. of 13 variables
r            15 obs. of 13 variables
Values
i            1
j            1
```

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearman
cov2cor(V)
```



```
r=data.germany
r=r-r
r$user=data.germany$user


for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){

      k=sort(a[,i],decreasing=TRUE)[2:6]
      l=order(-a[,i])[2:6]
      h=as.numeric(data.germany[l,(j+1)])

      r[i,(j+1)]=sum(h*k)/sum(k)
    }

reco=matrix(0,nrow=15,ncol=5)
```

```
C:/Users/Dell5/Desktop/Week7/Session 2 and 3/
> a[,i]
 [1] 1.0000000 0.2434400 0.6000000 0.2434400 0.7549704 0.2434400 0.1225148
 [8] 0.2434400 0.2434400 0.7827999 0.2000000 0.1225148 0.2000000 0.1612007
[15] 0.2434400
> sort(a[,i],decreasing=TRUE)
 [1] 1.0000000 0.7827999 0.7549704 0.6000000 0.2434400 0.2434400 0.2434400
 [8] 0.2434400 0.2434400 0.2434400 0.2000000 0.2000000 0.1612007 0.1225148
[15] 0.1225148
> sort(a[,i],decreasing=TRUE)[2:6]
[1] 0.7827999 0.7549704 0.6000000 0.2434400 0.2434400
```

```
a            15 obs. of 15 variables
data.germa.. 15 obs. of 13 variables
r            15 obs. of 13 variables
Values
  i          1
  j          1
```

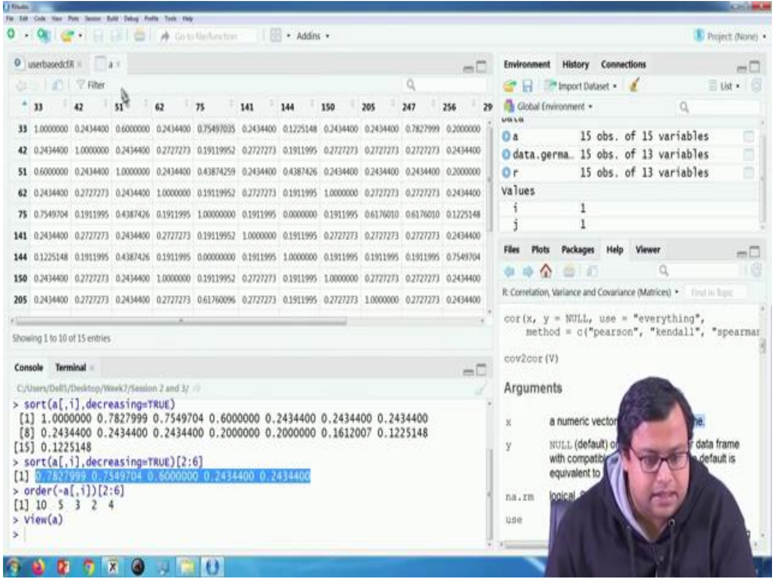
```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearman
cov2cor(V)
```

If i =1, j =1, what is a , i? That means a is ith column. This is the correlation values, i =1, that is why his correlation with himself is the highest, no issues in that and then these are the other correlations that this person has. Fair enough.

Now, if I sort this in decreasing order, that means from highest to lowest, I get this 1 comes at the top and then other values. Now, I will not take 1 because this is corresponding to himself, I will take 2 , 6, the next 5 based so, these are the top five similarity scores for ith customer with its neighbors, that means with his most similar customers so, these are my similarity score with me and my more similar customers. Let us say there are five professors who are most similar to me in terms of publishing certain papers in certain journals, whatever I publish, they also publish.

So, there are five such professors who find who are matching my interest level, my journal publication history, blah blah blah. So, they are most closest and this scores are those closeness score. Now, what is next? I will actually find out their order also. So, I will find out who they are. So, these are the purchase, this is the basically a , i to , c, that means this is actually their corresponding column numbers. So, the closest is 10, see here it is written if it is 8th, 9th, this is the 10, this original, original matrix of a i, original matrix of a i.

(Refer Slide Time: 20:34)



userbasedcf.R a

	33	42	51	62	75	141	144	150	205	247	256	29
33	1.0000000	0.2434400	0.6000000	0.2434400	0.75497035	0.2434400	0.1225148	0.2434400	0.2434400	0.7827999	0.2000000	
42	0.2434400	1.0000000	0.2434400	0.2727273	0.19119952	0.2727273	0.1911995	0.2727273	0.2727273	0.2727273	0.2434400	
51	0.6000000	0.2434400	1.0000000	0.2434400	0.43874259	0.2434400	0.4387426	0.2434400	0.2434400	0.2434400	0.2000000	
62	0.2434400	0.2727273	0.2434400	1.0000000	0.19119952	0.2727273	0.1911995	1.0000000	0.2727273	0.2727273	0.2434400	
75	0.7549704	0.1911995	0.4387426	0.1911995	1.00000000	0.1911995	0.0000000	0.1911995	0.6176010	0.6176010	0.1225148	
141	0.2434400	0.2727273	0.2434400	0.2727273	0.19119952	1.0000000	0.1911995	0.2727273	0.2727273	0.2727273	0.2434400	
144	0.1225148	0.1911995	0.4387426	0.1911995	0.00000000	0.1911995	1.0000000	0.1911995	0.1911995	0.1911995	0.7549704	
150	0.2434400	0.2727273	0.2434400	1.0000000	0.19119952	0.2727273	0.1911995	1.0000000	0.2727273	0.2727273	0.2434400	
205	0.2434400	0.2727273	0.2434400	0.2727273	0.61760096	0.2727273	0.1911995	0.2727273	1.0000000	0.2727273	0.2434400	

Showing 1 to 10 of 15 entries

```
C:/Users/Dell5/Desktop/Week7/Session 2 and 3/
> sort(a[,i],decreasing=TRUE)
 [1] 1.0000000 0.7827999 0.7549704 0.6000000 0.2434400 0.2434400 0.2434400
 [8] 0.2434400 0.2434400 0.2434400 0.2000000 0.2000000 0.1612007 0.1225148
[15] 0.1225148
> sort(a[,i],decreasing=TRUE)[2:6]
[1] 0.7827999 0.7549704 0.6000000 0.2434400 0.2434400
> order(-a[,i])[2:6]
[1] 10  5  3  2  4
> View(a)
>
```

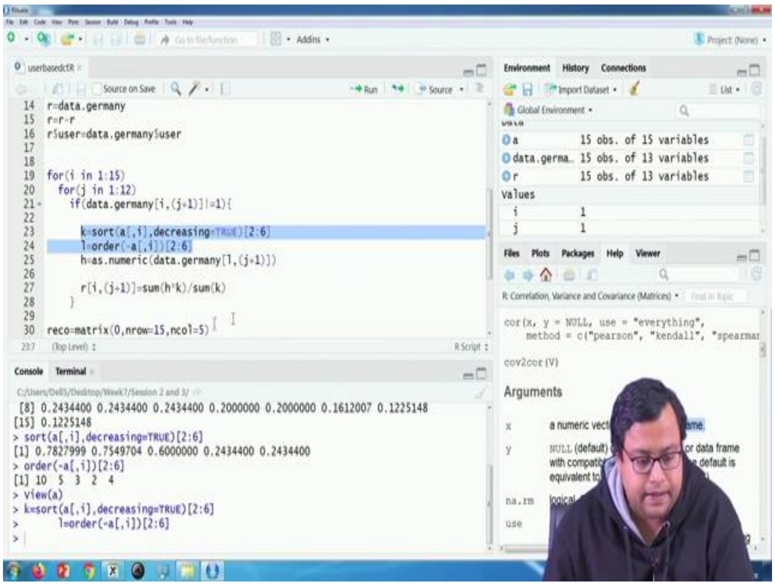
Environment: Global Environment

```
Data
a           15 obs. of 15 variables
data.germa..15 obs. of 13 variables
r           15 obs. of 13 variables
Values
i           1
j           1
```

R: Correlation, Variance and Covariance (Matrices)

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearmar
cov2cor(V)
```

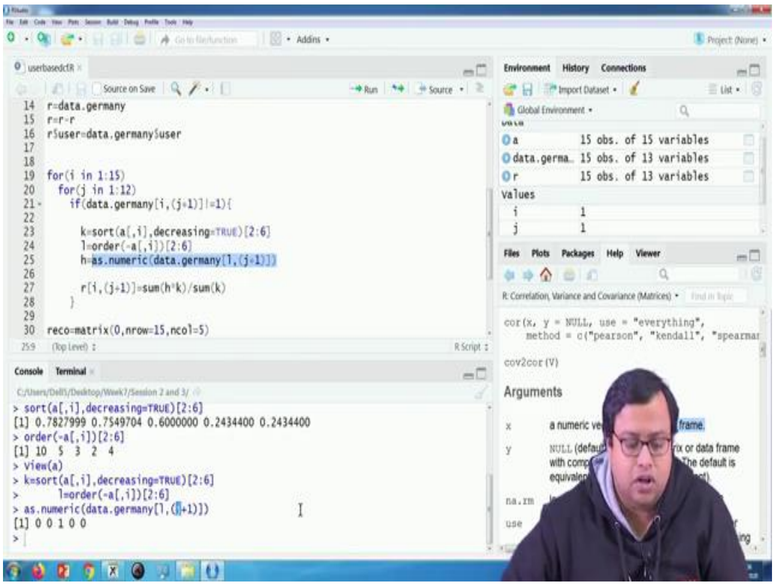
Arguments



userbasedcf.R

```
r=data.germany
r=r-r
r$user=data.germany$user


for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){

      k=sort(a[,i],decreasing=TRUE)[2:6]
      l=order(-a[,i])[2:6]
      h=as.numeric(data.germany[l,(j+1)])

      r[i,(j+1)]=sum(h*k)/sum(k)
    }

reco=matrix(0,nrow=15,ncol=5)
```

```
C:/Users/Dell5/Desktop/Week7/Session 2 and 3/
 [8] 0.2434400 0.2434400 0.2434400 0.2000000 0.2000000 0.1612007 0.1225148
[15] 0.1225148
> sort(a[,i],decreasing=TRUE)[2:6]
[1] 0.7827999 0.7549704 0.6000000 0.2434400 0.2434400
> order(-a[,i])[2:6]
[1] 10  5  3  2  4
> View(a)
> k=sort(a[,i],decreasing=TRUE)[2:6]
>       l=order(-a[,i])[2:6]
>
```



userbasedcf.R

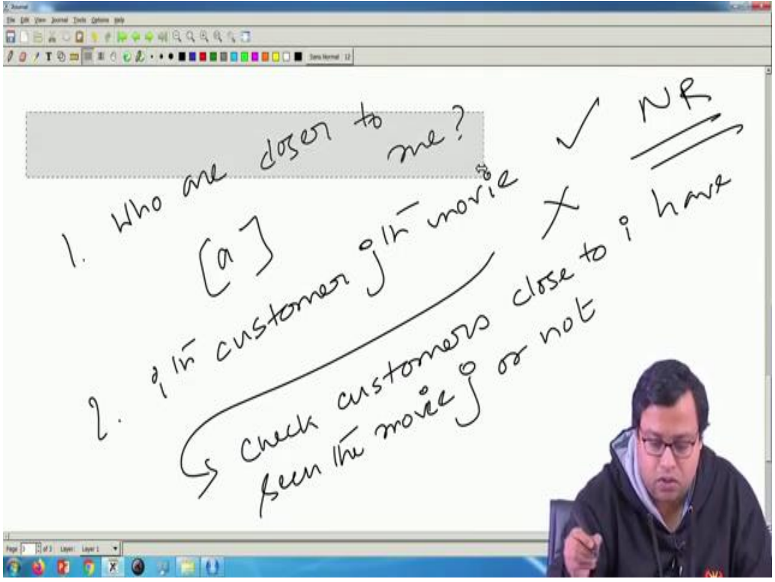
```
r=data.germany
r=r-r
r$user=data.germany$user


for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){

      k=sort(a[,i],decreasing=TRUE)[2:6]
      l=order(-a[,i])[2:6]
      h=as.numeric(data.germany[l,(j+1)])

      r[i,(j+1)]=sum(h*k)/sum(k)
    }

reco=matrix(0,nrow=15,ncol=5)
```

```
C:/Users/Dell5/Desktop/Week7/Session 2 and 3/
> sort(a[,i],decreasing=TRUE)[2:6]
[1] 0.7827999 0.7549704 0.6000000 0.2434400 0.2434400
> order(-a[,i])[2:6]
[1] 10  5  3  2  4
> View(a)
> k=sort(a[,i],decreasing=TRUE)[2:6]
>       l=order(-a[,i])[2:6]
> as.numeric(data.germany[l,(j+1)])
[1] 0 0 1 0 0
>
```

If I just check this first column, that 10th column is 782779 which is the highest, that value is coming here and corresponding I would say position is coming here. Similarly, the second best is fifth column, 0.7549 that value is coming here and correspondingly the position is coming here. So, that these are the positions and these are the similarity matrices. So, l is the position and then what is h? Now, this is something which you have to understand, what is h? h is the historical purchase data of this lth person, l persons. So, I will say what is h? If I just copy this, this part and paste it here l is basically which are the movies that these people have seen.

So, these five [l,(j+1)], that whether these guys have seen this j + 1th movie or not.

(Refer Slide Time: 21:55)

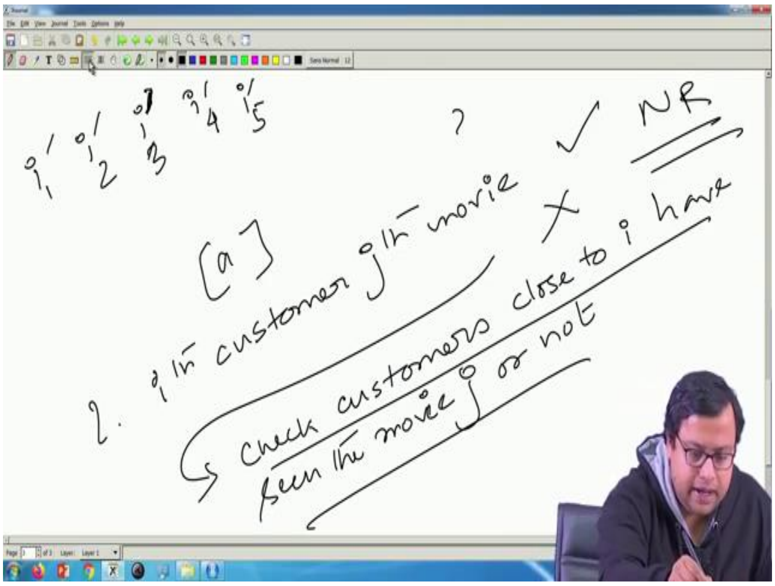


1. Who are closer to me? ✓ NR

[a]

2. i^{th} customer j^{th} movie ✗

↳ Check customers close to i have seen the movie j or not



i_1 i_2 i_3 i_4 i_5 ? ✓ NR

[a]

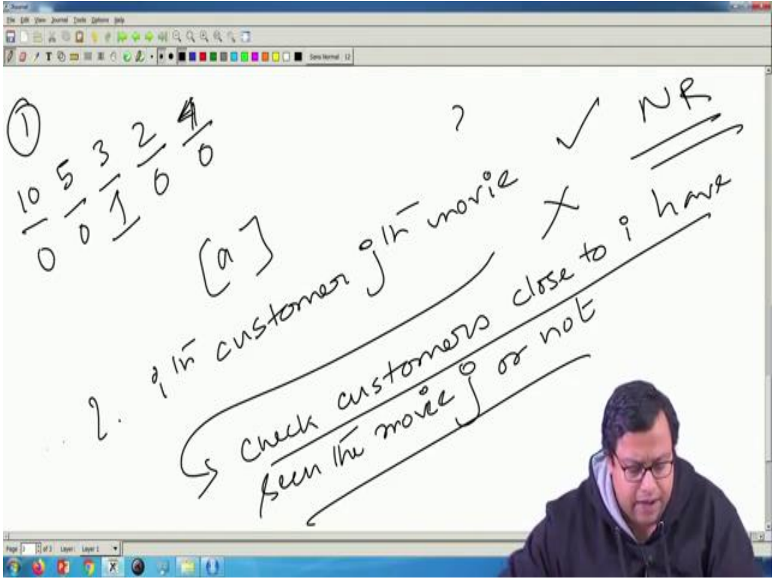
2. i^{th} customer j^{th} movie ✗

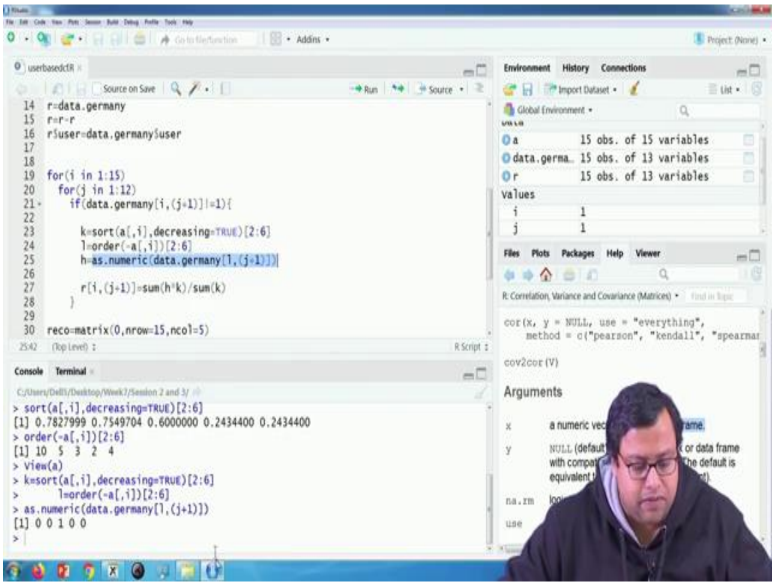
↳ Check customers close to i have seen the movie j or not

So, understand carefully what I am trying to do. I am finding out the first step, who are closer to me, that is my first question, that comes in the a data frame, fair enough. Then I check that, ith customer jth movie whether he has already seen, if he has already seen, no recommendation, if he has not seen then further analysis, what is the further analysis? Check costumers close to i have seen the movie or not, movie j or not so, whether customers close to i have seen movie j or not. So, that is my next question. So, whether customers who are close to me have seen this particular movie or not.

So, if they have seen this movie, then I will see this movie. If none of them have seen the movie then I will not see the movie. So, let us say who also this question, if I have to answer this question then first I have to find out who are the customers close to me, let us say customer I'_1, I'_2, I'_3, I'_4or I'_5 these are the customers who are closest to me. So in the case of customer number one, in the case of customer number one, the closest customer if you remember are customer number ten, then customer number five, then customer number ten, five, three, two, four.

(Refer Slide Time: 24:05)

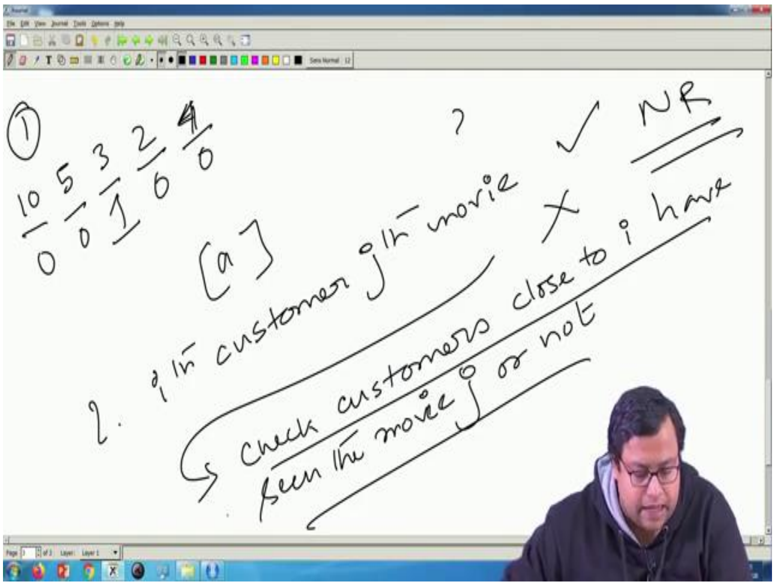


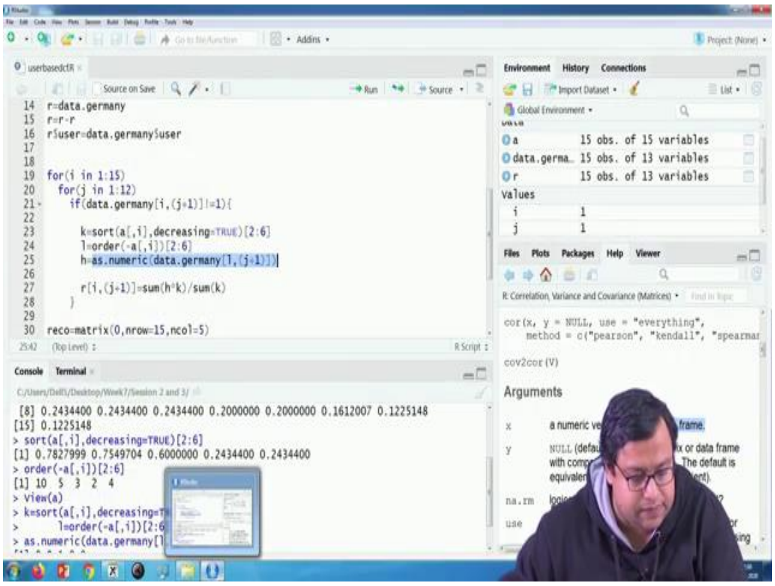


```
r=data.germany
r=r-r
r$user=data.germany$user


for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){

      k=sort(a[,i],decreasing=TRUE)[2:6]
      l=order(-a[,i])[2:6]
      h=as.numeric(data.germany[l,(j+1)])

      r[i,(j+1)]=sum(h*k)/sum(k)
    }

reco=matrix(0,nrow=15,ncol=5)
```

```
> sort(a[,i],decreasing=TRUE)[2:6]
[1] 0.7827999 0.7549704 0.6000000 0.2434400 0.2434400
> order(-a[,i])[2:6]
[1] 10  5  3  2  4
> View(a)
> k=sort(a[,i],decreasing=TRUE)[2:6]
>       l=order(-a[,i])[2:6]
> as.numeric(data.germany[l,(j+1)])
[1] 0 0 1 0 0
>
```

```
a             15 obs. of 15 variables
data.germa..  15 obs. of 13 variables
r             15 obs. of 13 variables
Values
  i           1
  j           1
```



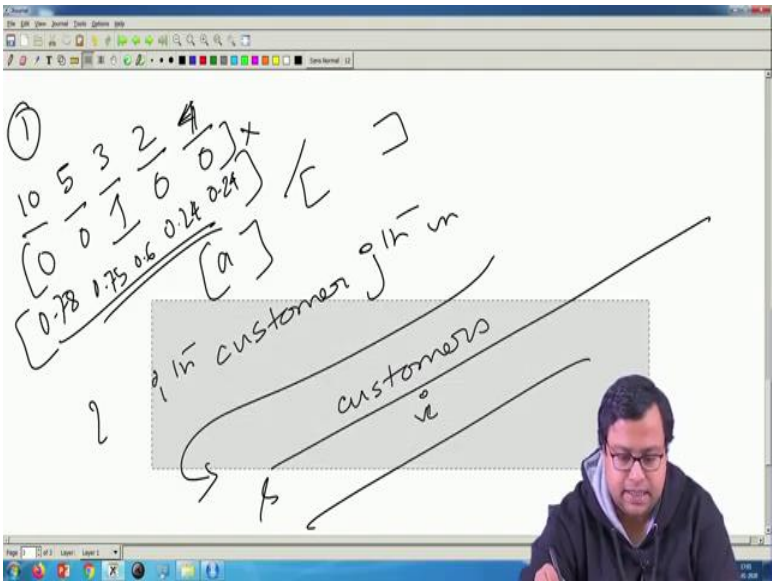


```
r=data.germany
r=r-r
r$user=data.germany$user


for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){

      k=sort(a[,i],decreasing=TRUE)[2:6]
      l=order(-a[,i])[2:6]
      h=as.numeric(data.germany[l,(j+1)])

      r[i,(j+1)]=sum(h*k)/sum(k)
    }

reco=matrix(0,nrow=15,ncol=5)
```

```
C:/Users/Dell1/Desktop/Week7/Session 2 and 3/
 [8] 0.2434400 0.2434400 0.2434400 0.2000000 0.2000000 0.1612007 0.1225148
[15] 0.1225148
> sort(a[,i],decreasing=TRUE)[2:6]
[1] 0.7827999 0.7549704 0.6000000 0.2434400 0.2434400
> order(-a[,i])[2:6]
[1] 10  5  3  2  4
> View(a)
> k=sort(a[,i],decreasing=T
>       l=order(-a[,i])[2:6
> as.numeric(data.germany[l
```

```
a          15 obs. of 15 variables
data.germa.. 15 obs. of 13 variables
r          15 obs. of 13 variables
Values
  i        1
  j        1
```

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearmar

cov2cor(V)
```



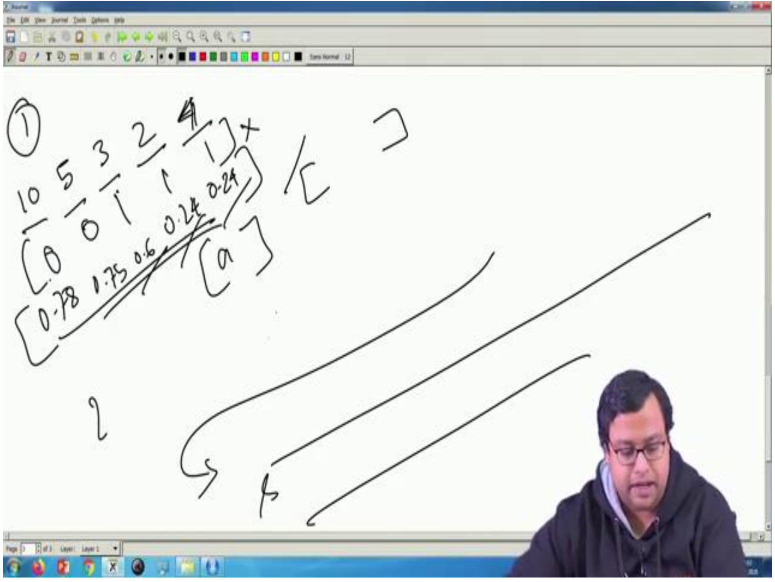
So these are if I am checking for customer number one, ten, five, three, two, four, these are the customers who are closest to me. Now, I have to check whether they have seen the jth movie or not, jth movie means the first movie. If they have seen the jth movie, then I will do something, if none of them seen the movie then there is no recommendation. So, let us say, that is what I find out the 10th person has not seen, this person has not seen, this person has not seen, this person has seen, this person has not seen, this person has not seen, how will I get this data?

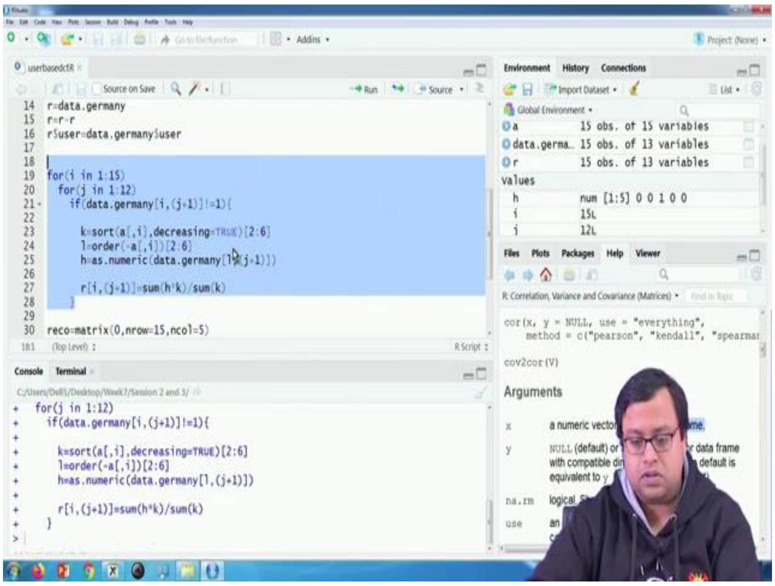
I will get this data from my data . Germany, see data . Germany l, l is the IDs, customer IDs, l row number, and j + 1 column number, why j + 1? Because first column was the user IDs. So, l , j + 1 that is giving me the sale values, while the column number is j + 1 and row number is l, now l has five entries so, all those five sale values will be given and I am changing into numeric value.

So, what is the numeric value? That numeric value is 0, 0, 1, 0, 0. That means that 10th guy has seen it, 5th guy has not seen it, 3rd guy has seen it, 2nd, 4th, 10th, 5th nobody else have seen and then what I am trying to calculate? I am trying to calculate the recommendation matrix, how strong strongly I should recommend? I should strongly recommend if see, 0.78, 0.75, 0.6 this is 0.78, this is 0.75, this is 0.6 and then come 0.24, 0.24 okay, 0.24, 0.24. So basically, I am trying to say that whether I should be recommended this particular movie or not will depend on the multiplication of this historical data and the, the k, which is the weightages. So this is my actual observation, this is the weightages so, create a weighted average.

So, this × this, this × this /∑all of these things. So, what I am getting basically, I am getting 1 × 0.6 at the top, and then summation of all these things in the numerator, by chance think about a situation, by chance if instead of this one person, let us say two persons have seen the movie, by chance two persons have seen the movie, how will you deal with that kind of a situation? Wait a minute, let me just delete this.

(Refer Slide Time: 27:14)



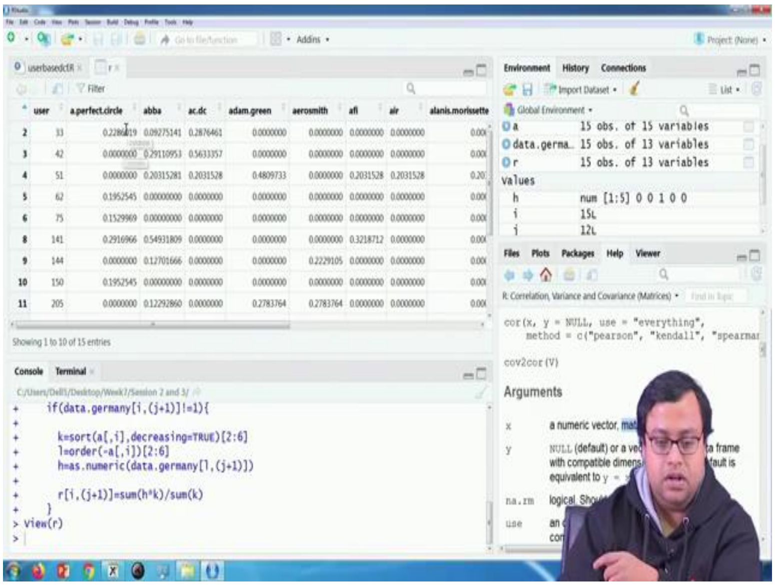


```
r=data.germany
r=r-r
r$user=data.germany$user


for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){

      k=sort(a[,i],decreasing=TRUE)[2:6]
      l=order(-a[,i])[2:6]
      h=as.numeric(data.germany[l,(j+1)])

      r[i,(j+1)]=sum(h*k)/sum(k)
    }

reco=matrix(0,nrow=15,ncol=5)
```

```
+   for(j in 1:12)
+     if(data.germany[i,(j+1)]!=1){
+
+       k=sort(a[,i],decreasing=TRUE)[2:6]
+       l=order(-a[,i])[2:6]
+       h=as.numeric(data.germany[l,(j+1)])
+
+       r[i,(j+1)]=sum(h*k)/sum(k)
+     }
>
```

```
a           15 obs. of 15 variables
data.germa.. 15 obs. of 13 variables
r           15 obs. of 13 variables
Values
h           num [1:5] 0 0 1 0 0
i           15L
j           12L
```

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearman"
```



	user	a.perfect.circle	abba	ac.dc	adam.green	aerosmith	afi	air	alanis.morissette
2	33	0.2286019	0.09275141	0.2876461	0.0000000	0.0000000	0.0000000	0.0000000	0.00
3	42	0.0000000	0.29110953	0.5633357	0.0000000	0.0000000	0.0000000	0.0000000	0.00
4	51	0.0000000	0.20315281	0.2031528	0.4809733	0.0000000	0.2031528	0.2031528	0.20
5	62	0.1952545	0.00000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.00
6	75	0.1529969	0.00000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.00
8	141	0.2916966	0.54931809	0.0000000	0.0000000	0.0000000	0.3218712	0.0000000	0.00
9	144	0.0000000	0.12701666	0.0000000	0.0000000	0.2229105	0.0000000	0.0000000	0.00
10	150	0.1952545	0.00000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.00
11	205	0.0000000	0.12292860	0.0000000	0.2783764	0.2783764	0.0000000	0.0000000	0.00

Showing 1 to 10 of 15 entries

```
+     if(data.germany[i,(j+1)]!=1){
+
+       k=sort(a[,i],decreasing=TRUE)[2:6]
+       l=order(-a[,i])[2:6]
+       h=as.numeric(data.germany[l,(j+1)])
+
+       r[i,(j+1)]=sum(h*k)/sum(k)
+     }
> View(r)
>
```

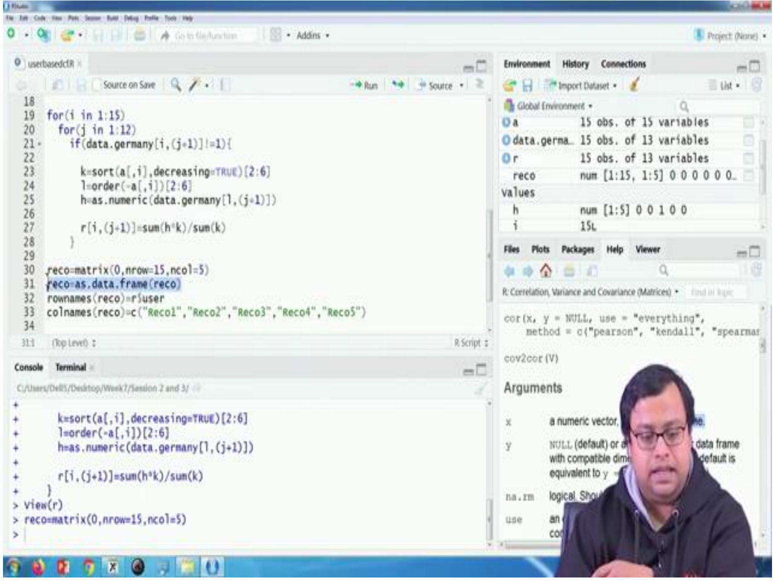
So, by chance, let us say, instead of this particular person, by chance, if this person was 1, that means that your closest friend, your closest user has seen the movie, obviously that should increase. On the other hand, let us say instead of this person seen the movie, this 1 × 0.78, what would have been the case if this is this was 0 but let us say this was 1, this was 1, this was 1? Then more people probably your closest person has not seen, but more number of people who are similar to you have seen, so that is why the numerator would have been 0.6 + 0.24 + 0.24.

So, that numerator is basically the measurement of the how strongly we should recommend and that is coming up from this particular thing. r [i , j + 1] is basically sum of h × k. h stands for the historical purchase history of users who were close to you. k is basically the similarity score by sum of k so, this is where I run for the whole thing. It is quick, it has given me some scores, zeroes mean either nobody who is close to me has seen these movies.

Remember in the item to item, whether I have seen Shahrukh Khan’s others movies, let us say you are trying to recommend me, you are deciding whether you should recommend Main Hoon Na to me, and you were checking that whether I have seen Main Hoon Na, if I have seen Main Hoon Na, no recommendation. If I have not seen Main Hoon Na then whether I have seen other movies which are close to Main Hoon Na, here story is different.

Here if I have not seen Main Hoon Na then you will see that whether other users who are close to me have seen Main Hoon Na so, that is what you are saying, all the zeros means no other users which are close to me, who are similar to me have seen Main Hoon Na, that is why it is coming zero or I have already seen that particular movie, that is why it is coming zero. Otherwise there is some score coming and those scores what will I do?

(Refer Slide Time: 29:29)



```
for(i in 1:15)
  for(j in 1:12)
    if(data.germany[i,(j+1)]!=1){

      k=sort(a[,i],decreasing=TRUE)[2:6]
      l=order(-a[,i])[2:6]
      h=as.numeric(data.germany[l,(j+1)])

      r[i,(j+1)]=sum(h*k)/sum(k)
    }

reco=matrix(0,nrow=15,ncol=5)
reco=as.data.frame(reco)
rownames(reco)=r$user
colnames(reco)=c("Reco1","Reco2","Reco3","Reco4","Reco5")
```

C:/Users/Dell5/Desktop/Week7/Session 2 and 3/

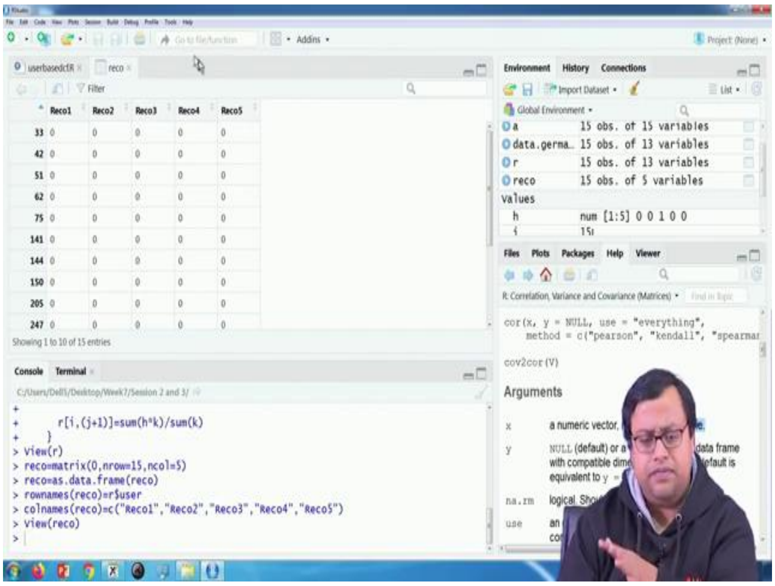
```
+
+      k=sort(a[,i],decreasing=TRUE)[2:6]
+      l=order(-a[,i])[2:6]
+      h=as.numeric(data.germany[l,(j+1)])
+
+      r[i,(j+1)]=sum(h*k)/sum(k)
+    }
> View(r)
> reco=matrix(0,nrow=15,ncol=5)
>
```

```
Global Environment
a            15 obs. of 15 variables
data.germa…  15 obs. of 13 variables
r            15 obs. of 13 variables
  reco       num [1:15, 1:5] 0 0 0 0 0 0…
Values
  h          num [1:5] 0 0 1 0 0
  i          15L
```

R: Correlation, Variance and Covariance (Matrices)

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearmar
cov2cor(V)
```

Arguments



	Reco1	Reco2	Reco3	Reco4	Reco5
33	0	0	0	0	0
42	0	0	0	0	0
51	0	0	0	0	0
62	0	0	0	0	0
75	0	0	0	0	0
141	0	0	0	0	0
144	0	0	0	0	0
150	0	0	0	0	0
205	0	0	0	0	0
247	0	0	0	0	0

Showing 1 to 10 of 15 entries

```
C:/Users/Dell5/Desktop/Week7/Session 2 and 3/
+
+       r[i,(j+1)]=sum(h*k)/sum(k)
+     }
> View(r)
> reco=matrix(0,nrow=15,ncol=5)
> reco=as.data.frame(reco)
> rownames(reco)=r$user
> colnames(reco)=c("Reco1","Reco2","Reco3","Reco4","Reco5")
> View(reco)
>
```

```
a             15 obs. of 15 variables
data.germa... 15 obs. of 13 variables
r             15 obs. of 13 variables
reco          15 obs. of 5 variables
Values
  h           num [1:5] 0 0 1 0 0
  i           15
```

R: Correlation, Variance and Covariance (Matrices)

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearman
cov2cor(V)
```

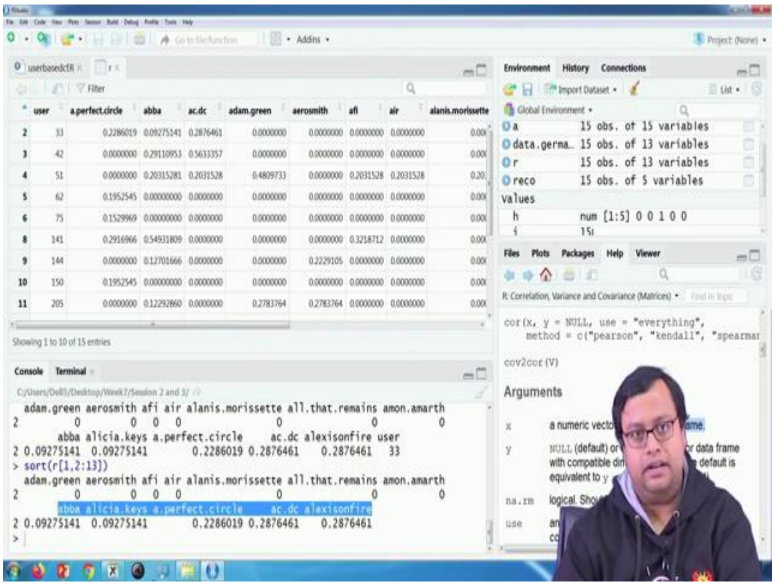
Arguments

x a numeric vector,

y NULL (default) or a ... data frame with compatible dim... default is equivalent to y = ...

na.rm logical. Sho...

use an ... co...



	user	a.perfect.circle	abba	ac.dc	adam.green	aerosmith	afi	air	alanis.morissette
2	33	0.2286019	0.09275141	0.2876461	0.0000000	0.0000000	0.0000000	0.0000000	0.000
3	42	0.0000000	0.29110953	0.5633357	0.0000000	0.0000000	0.0000000	0.0000000	0.000
4	51	0.0000000	0.20315281	0.2031528	0.4809733	0.0000000	0.2031528	0.2031528	0.20
5	62	0.1952545	0.00000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.000
6	75	0.1529969	0.00000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.000
8	141	0.2916966	0.54931809	0.0000000	0.0000000	0.0000000	0.3218712	0.0000000	0.000
9	144	0.0000000	0.12701666	0.0000000	0.0000000	0.2229105	0.0000000	0.0000000	0.000
10	150	0.1952545	0.00000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.000
11	205	0.0000000	0.12292860	0.0000000	0.2783764	0.2783764	0.0000000	0.0000000	0.000

Showing 1 to 10 of 15 entries

```
C:/Users/Dell5/Desktop/Week7/Session 2 and 3/
  adam.green aerosmith afi air alanis.morissette all.that.remains amon.amarth
2          0         0   0   0                 0                0           0
        abba alicia.keys a.perfect.circle     ac.dc alexisonfire user
2 0.09275141  0.09275141        0.2286019 0.2876461    0.2876461   33
> sort(r[1,2:13])
  adam.green aerosmith afi air alanis.morissette all.that.remains amon.amarth
2          0         0   0   0                 0                0           0
        abba alicia.keys a.perfect.circle     ac.dc alexisonfire
2 0.09275141  0.09275141        0.2286019 0.2876461    0.2876461
>
```

```
a             15 obs. of 15 variables
data.germa... 15 obs. of 13 variables
r             15 obs. of 13 variables
reco          15 obs. of 5 variables
Values
  h           num [1:5] 0 0 1 0 0
  i           15
```

R: Correlation, Variance and Covariance (Matrices)

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearman
cov2cor(V)
```

Arguments

x a numeric vecto... ame.

y NULL (default) or ... r data frame with compatible di... default is equivalent to y ...

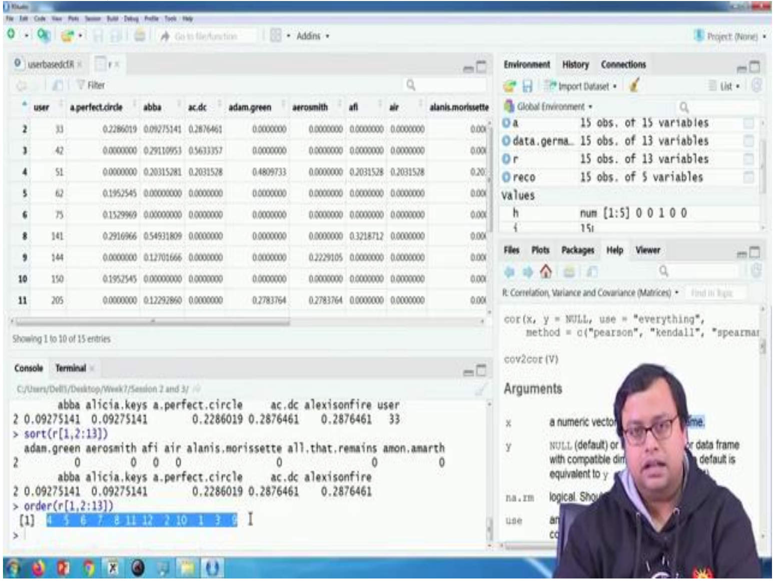
na.rm logical. Sho...

use an ... co...

Simple, those score is something is as usual, I will find out the row names, column names and so I am creating a Reco matrix, which is all my usernames and Reco 1, Reco 2, Reco 3, Reco 4, Reco 5, just five column names. And here I will populate, what will I populate? I will populate the column names, I will pick up one particular so, from this r column if I am doing it for the first column let us say.

So, r , 1, sorry 1 ,, this is what it looks like. Fair enough. Now, if I sort it up, this is how it looks like, the lower is zero, the higher is this number. If I sort it up not everything, but let us say r has 13 variable so 2 to 13 then the user, this one is vanished, this user one is vanished, last column. So, I get these are basically the recommendation scores. So which one I will recommend? These five basically I will recommend, the last five. So what I have done, I am doing an order, instead of sorting up I am doing an order.

(Refer Slide Time: 30:54)



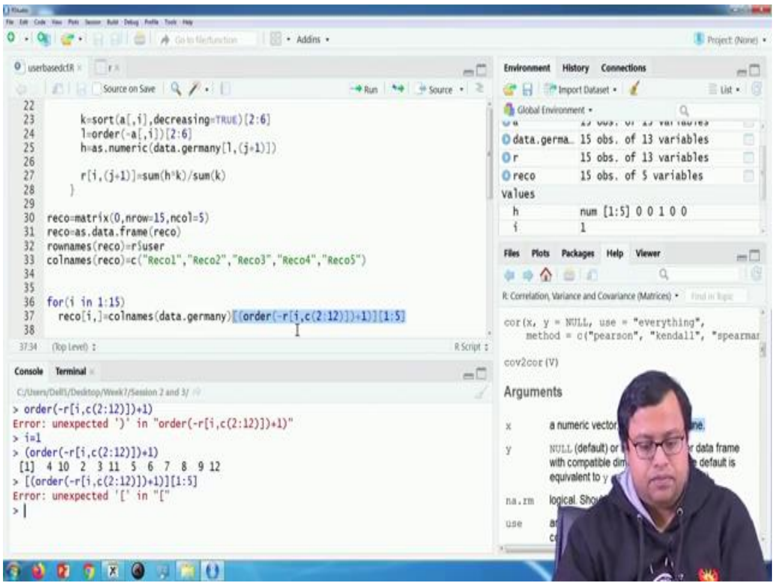
	user	a.perfect.circle	abba	ac.dc	adam.green	aerosmith	afi	air	alanis.morissette
2	33	0.2286019	0.09275141	0.2876461	0.0000000	0.0000000	0.0000000	0.0000000	0.00(
3	42	0.0000000	0.29110953	0.5633357	0.0000000	0.0000000	0.0000000	0.0000000	0.00(
4	51	0.0000000	0.20315281	0.2031528	0.4809733	0.0000000	0.2031528	0.2031528	0.20:
5	62	0.1952545	0.00000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.00(
6	75	0.1529969	0.00000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.00(
8	141	0.2916966	0.54931809	0.0000000	0.0000000	0.0000000	0.3218712	0.0000000	0.00(
9	144	0.0000000	0.12701666	0.0000000	0.0000000	0.2229105	0.0000000	0.0000000	0.00(
10	150	0.1952545	0.00000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.00(
11	205	0.0000000	0.12292860	0.0000000	0.2783764	0.2783764	0.0000000	0.0000000	0.00(

Showing 1 to 10 of 15 entries

```
       abba alicia.keys a.perfect.circle     ac.dc alexisonfire user
2 0.09275141  0.09275141        0.2286019 0.2876461    0.2876461   33
> sort(r[1,2:13])
  adam.green aerosmith afi air alanis.morissette all.that.remains amon.amarth
2          0         0   0   0                 0                0           0
       abba alicia.keys a.perfect.circle     ac.dc alexisonfire
2 0.09275141  0.09275141        0.2286019 0.2876461    0.2876461
> order(r[1,2:13])
 [1]  4  5  6  7  8 11 12  2 10  1  3  9
>
```

```
a            15 obs. of 15 variables
data.germa.. 15 obs. of 13 variables
r            15 obs. of 13 variables
reco         15 obs. of 5 variables
Values
h            num [1:5] 0 0 1 0 0
i            15
```

```
cor(x, y = NULL, use = "everything",
    method = c("pearson", "kendall", "spearman
```



```
        k=sort(a[,i],decreasing=TRUE)[2:6]
        l=order(-a[,i])[2:6]
        h=as.numeric(data.germany[1,(j+1)])

        r[i,(j+1)]=sum(h*k)/sum(k)
      }

reco=matrix(0,nrow=15,ncol=5)
reco=as.data.frame(reco)
rownames(reco)=r$user
colnames(reco)=c("Reco1","Reco2","Reco3","Reco4","Reco5")


for(i in 1:15)
  reco[i,]=colnames(data.germany)[(order(-r[i,c(2:12)])+1)][1:5]

```

```
> order(-r[i,c(2:12)])+1)
Error: unexpected ')' in "order(-r[i,c(2:12)])+1)"
> i=1
> (order(-r[i,c(2:12)])+1)
 [1]  4 10  2  3 11  5  6  7  8  9 12
> [(order(-r[i,c(2:12)])+1)][1:5]
Error: unexpected '[' in "["
> |
```

An order is giving me the basically the position of this lowest to highest and etcetera and I am using that position to find out the column names, so order of this is giving me the position here, when let us say i =1, when i =1, the order is basically giving me the position.

And I am taking the top five, because I will recommend only five. So, I am taking the top five of them. And then with that top five, let me come back. So, I am taking up the column names of data . Germany. So, you give me the column names of data . Germany for those specific positions. So, these are basically for first customer, these are the suggestions but I will take the top five, that is 1 to 5. So, this is the suggestion for i = 1.

Similarly, i =2 it will change, i =3 it will change. So I am just running that in a loop. So if I just run this, the recommendation table is populated now. The first one is the top five based on the r matrix. The second one is the top five based on the second row of r matrix and so on. And that is how you are creating recommendation engine which is user base. So what is the difference basically? Once more it is item to item similarity that we are checking in item based collaborative filtering, this user to user similarity that we are checking in user base in similarity matrix.

We will do some more examples in the next videos. Thank you for being with me, I will strongly suggest that you should go ahead and check and run these codes once more with the bigger dataset. It might take some time but you will find out more nuances, sometimes some errors also, let me know whether you are getting stuck in somewhere and we will try to solve that. Thank you very much for listening to this video. I will see you in the next video.