Final Project

Steve Dunn

July 17, 2016

Prof. Andy Catlin IS 643 - Special Topics: Recommender Systems Department of Data Analytics, City University of New York

For my Final project I will build a recommendation system that will look at recommendation from Item Based Collaborative Filtering, User Based Collaborative Filtering and recommendations by genre with Hierarchal clustering. I will create models to get Movies from similar Genres based on the distance of the clusters.Hierarchal Clustering is an unsupervised learning method where the goal is to segment data into similar groups.The dataset is taken from the MovieLens dataset avaliable in the Recommenderlab package and from their website http://grouplens.org/datasets/movielens.

Data Exploration

```r
#Visualization of the data
str(MovieLense)
```

```
## Formal class 'realRatingMatrix' [package "recommenderlab"] with 2 slots
##   ..@ data     :Formal class 'dgCMatrix' [package "Matrix"] with 6 slots
##   .. .. ..@ i       : int [1:99392] 0 1 4 5 9 12 14 15 16 17 ...
##   .. .. ..@ p       : int [1:1665] 0 452 583 673 882 968 994 1386 1605 1904 ...
##   .. .. ..@ Dim     : int [1:2] 943 1664
##   .. .. ..@ Dimnames:List of 2
##   .. .. .. ..$ : chr [1:943] "1" "2" "3" "4" ...
##   .. .. .. ..$ : chr [1:1664] "Toy Story (1995)" "GoldenEye (1995)" "Four Rooms (1995)" "Get Shorty
##   .. .. ..@ x       : num [1:99392] 5 4 4 4 4 3 1 5 4 5 ...
##   .. .. ..@ factors : list()
##   ..@ normalize: NULL
```

```r
class(MovieLense)
```

```
## [1] "realRatingMatrix"
## attr(,"package")
## [1] "recommenderlab"
```

```r
head(MovieLense)
```

```
## 1 x 1664 rating matrix of class 'realRatingMatrix' with 271 ratings.
```

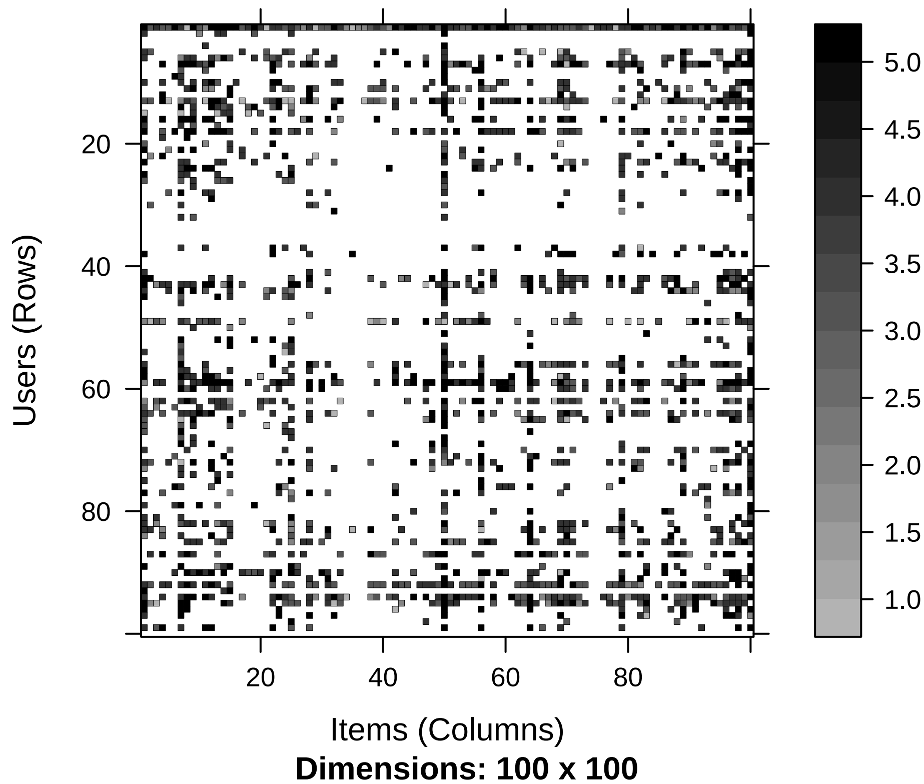
```r
dim(MovieLense)
```

```
## [1]  943 1664
```

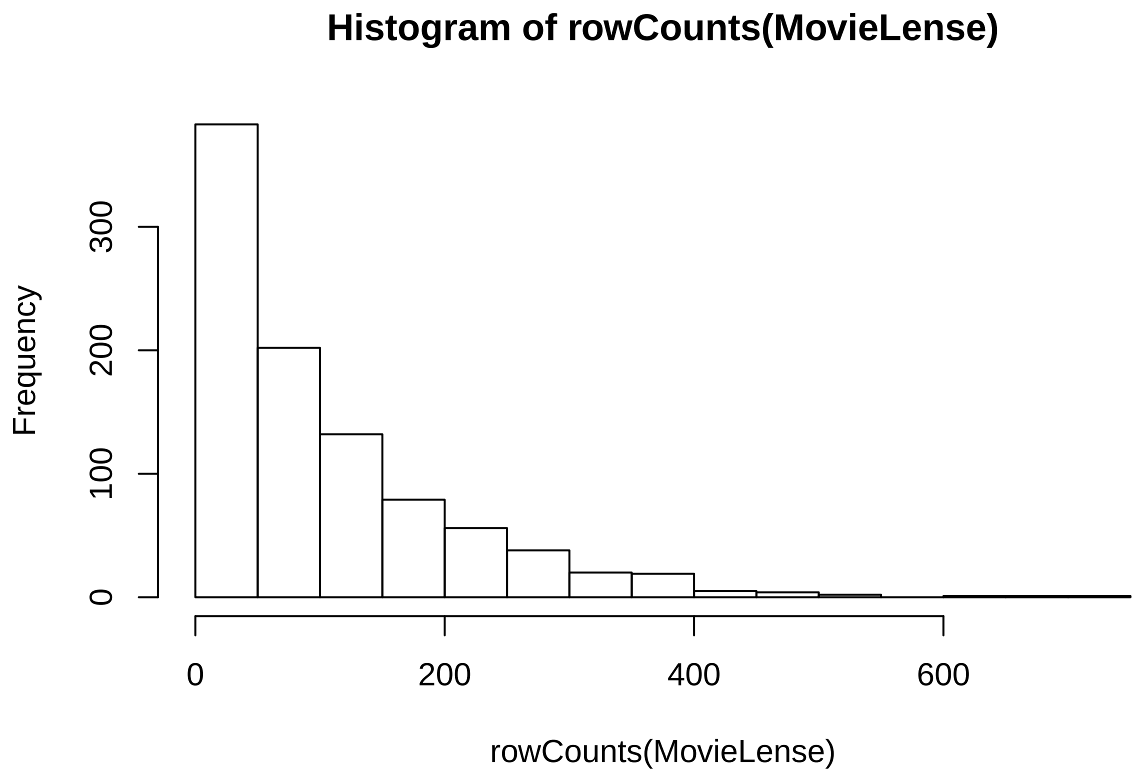
```
summary(MovieLense)
```

```
##          Length            Class            Mode
##               1 realRatingMatrix              S4
```

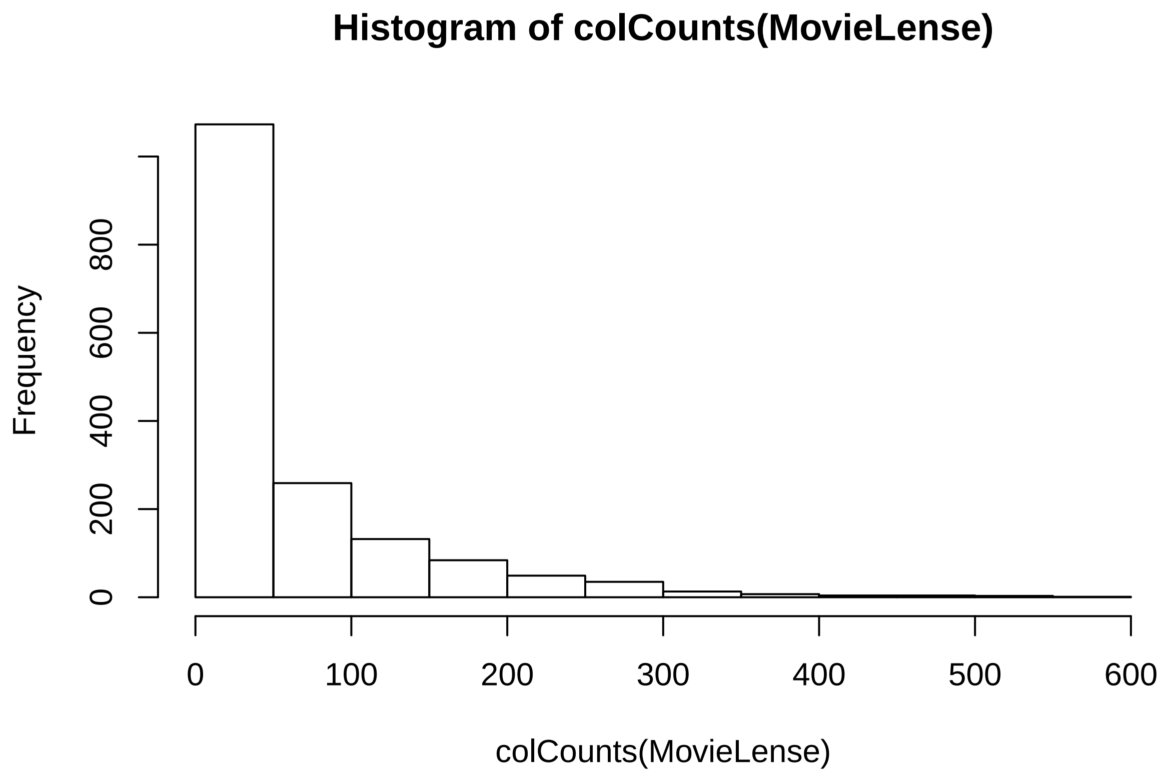
```
## visualize part of the matrix
image(MovieLense[1:100,1:100])
```




```
## number of ratings per user
hist(rowCounts(MovieLense))
```




```
## number of ratings per movie
hist(colCounts(MovieLense))
```




```
## mean rating (averaged over users)
mean(rowMeans(MovieLense))
```

```
## [1] 3.587565
```

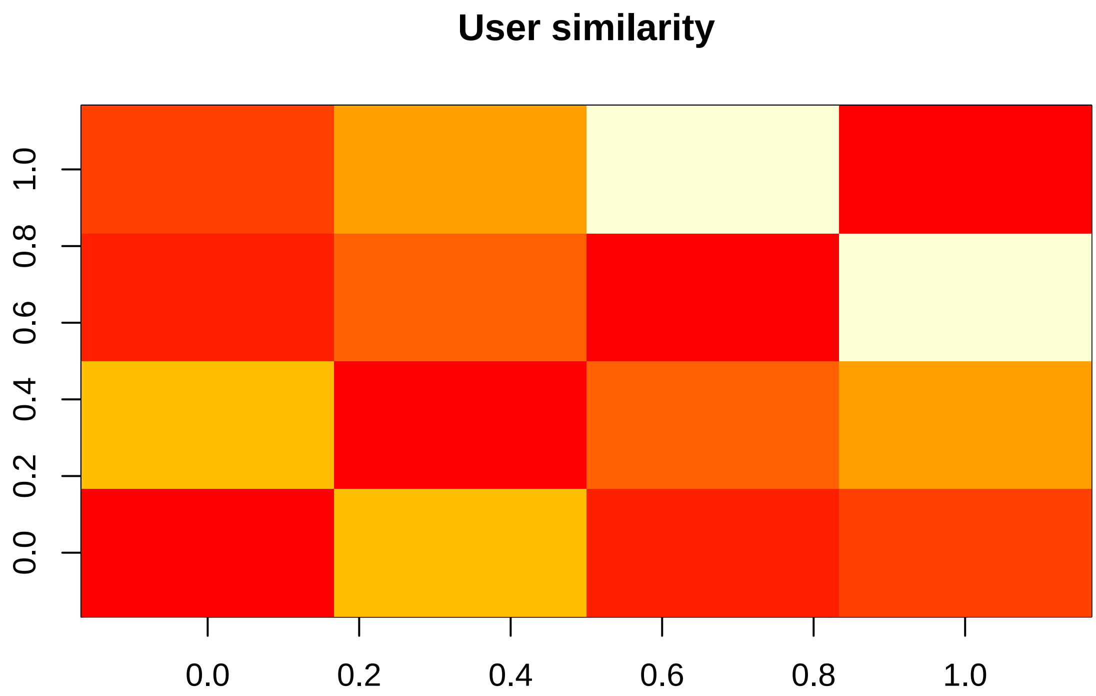
calculate similarity between user ratings and item ratings

the red areas denote the similarity of the ratings,the diagonals are all red because it is measured against it self.There seems to be more similarity between user ratings than there is between items.

```
#calculate Similarity matrix
similarity_users <- similarity(MovieLense[1:4,], method = "cosine", which = "users")
as.matrix( similarity_users)
```

```
##            1          2          3          4
## 1 0.00000000 0.16893670 0.03827203 0.06634975
## 2 0.16893670 0.00000000 0.09706862 0.15310468
## 3 0.03827203 0.09706862 0.00000000 0.33343036
## 4 0.06634975 0.15310468 0.33343036 0.00000000
```

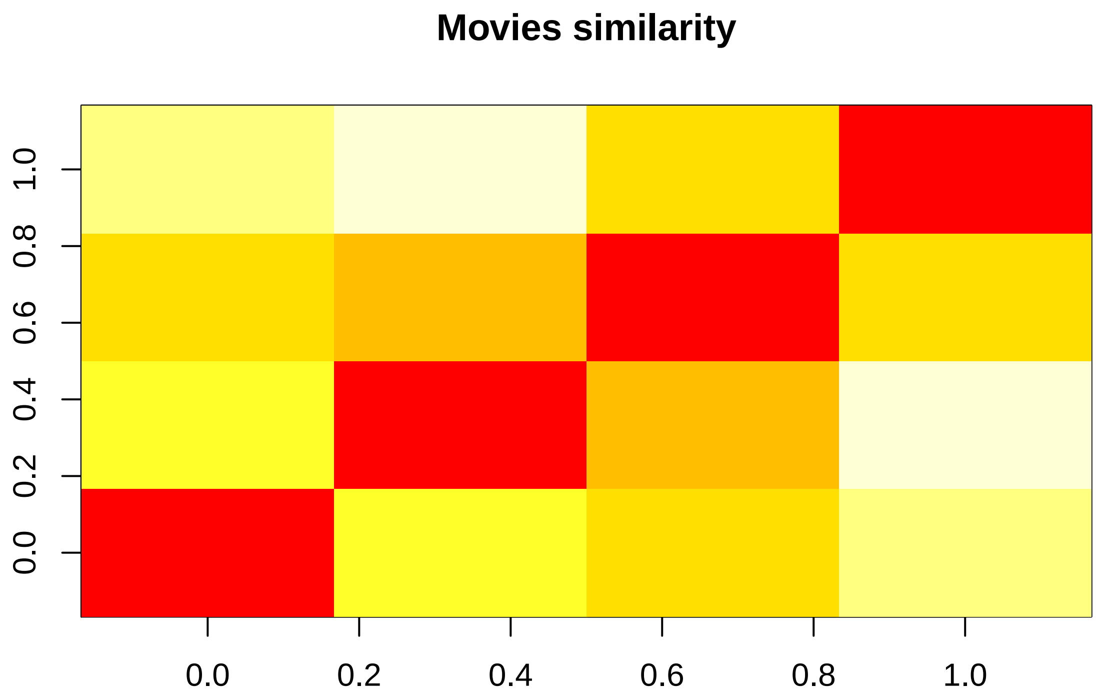
```
image(as.matrix(similarity_users), main = "User similarity")
```




```
  similarity_items <- similarity(MovieLense[,1:4], method = "cosine", which = "items")
as.matrix(similarity_items)
```

```
##                   Toy Story (1995) GoldenEye (1995) Four Rooms (1995)
## Toy Story (1995)         0.0000000        0.4023822         0.3302448
## GoldenEye (1995)         0.4023822        0.0000000         0.2730692
## Four Rooms (1995)        0.3302448        0.2730692         0.0000000
## Get Shorty (1995)        0.4549379        0.5025708         0.3248664
##                   Get Shorty (1995)
## Toy Story (1995)          0.4549379
## GoldenEye (1995)          0.5025708
## Four Rooms (1995)         0.3248664
## Get Shorty (1995)         0.0000000
```

```
image(as.matrix(similarity_items), main = "Movies similarity")
```



splitting the data set into training and test where i will use the training set to rate the data and use the test data set to predict the movies by IBCF and UBCF

```
#==============================================================================================================
#Splitting the dataset into Train and test
#==============================================================================================================
#Subsetting the dataset
ratings_movies <- MovieLense[ rowCounts( MovieLense) > 50, colCounts( MovieLense) > 100]
ratings_norm<-(ratings_movies)

which_train<-sample(x=c(TRUE,FALSE),size = nrow(ratings_norm),replace = TRUE,prob = c(.8,.2))
rec_data_train<-ratings_movies[which_train,]
rec_data_test<-ratings_movies[!which_train,]
```

Item Based Collaborative Filtering

```
#Prediction for Item Based Collaborative Filtering

#build recommender Model
rec_modelI<-Recommender(data=rec_data_train,method='IBCF')

#predict 5 movies for a specific users
n_rec<-5
recom2<-predict(rec_modelI,newdata=rec_data_test[1],n=n_rec)
#convert recommenderlab object to readable list
```

```r
recom_list <- as(recom2, "list")
recom_list
```

```
## $`1`
## [1] "L.A. Confidential (1997)"         "In the Name of the Father (1993)"
## [3] "Schindler's List (1993)"          "Boot, Das (1981)"
## [5] "Rear Window (1954)"
```

```r
#predict 5 movies for users
rec_predictI<-predict(rec_modelI,newdata=rec_data_test,n=n_rec)
rec_predictI
```

```
## Recommendations as 'topNList' with n = 5 for 107 users.
```

```r
rec_matrix<-sapply (rec_predictI@items,
function(x)
  {
  colnames(rec_data_test)[x]
  }
)
rec_matrix[,1:5]
```

```
##      1
## [1,] "L.A. Confidential (1997)"
## [2,] "In the Name of the Father (1993)"
## [3,] "Schindler's List (1993)"
## [4,] "Boot, Das (1981)"
## [5,] "Rear Window (1954)"
##      3
## [1,] "Ed Wood (1994)"
## [2,] "Willy Wonka and the Chocolate Factory (1971)"
## [3,] "Clockwork Orange, A (1971)"
## [4,] "Donnie Brasco (1997)"
## [5,] "Heathers (1989)"
##      10                                    23
## [1,] "Truth About Cats & Dogs, The (1996)" "Big Night (1996)"
## [2,] "To Kill a Mockingbird (1962)"        "Wrong Trousers, The (1993)"
## [3,] "Birds, The (1963)"                   "Boot, Das (1981)"
## [4,] "Godfather: Part II, The (1974)"      "Trainspotting (1996)"
## [5,] "Full Metal Jacket (1987)"            "Graduate, The (1967)"
##      28
## [1,] "What's Eating Gilbert Grape (1993)"
## [2,] "Leaving Las Vegas (1995)"
## [3,] "Being There (1979)"
## [4,] "True Romance (1993)"
## [5,] "Trainspotting (1996)"
```

User Based Collaborative Filtering

```r
#===========================================================================================
#Prediction for User Based Collaborative Filtering
```

```r
#===================================================================================
#build recommender Model
rec_modelU<-Recommender(data=rec_data_train,method='UBCF')
model_details <- getModel(rec_modelU)

#predict 5 movies for specific users
n_rec<-10

#Obtain 10 recommendations for 1st user in dataset
recom <- predict(object=rec_modelU, newdata=rec_data_test[1], n=n_rec)

#output ot a list
recom_list <- as(recom, "list")
recom_list
```

```
## $`1`
##  [1] "Glory (1989)"
##  [2] "Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1963)"
##  [3] "Close Shave, A (1995)"
##  [4] "Lawrence of Arabia (1962)"
##  [5] "Casablanca (1942)"
##  [6] "L.A. Confidential (1997)"
##  [7] "Harold and Maude (1971)"
##  [8] "Butch Cassidy and the Sundance Kid (1969)"
##  [9] "Rear Window (1954)"
## [10] "Magnificent Seven, The (1954)"
```

```r
#predict  movies for users
rec_predictU<-predict(rec_modelU,newdata=rec_data_test,n=n_rec)
rec_predictU
```

```
## Recommendations as 'topNList' with n = 10 for 107 users.
```

```r
rec_matrixU<-sapply(rec_predictU@items,
                    function(x){
                      colnames(rec_data_test)[x]
                      })
rec_matrixU[,1:5]
```

```
##       1
##  [1,] "Glory (1989)"
##  [2,] "Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1963)"
##  [3,] "Close Shave, A (1995)"
##  [4,] "Lawrence of Arabia (1962)"
##  [5,] "Casablanca (1942)"
##  [6,] "L.A. Confidential (1997)"
##  [7,] "Harold and Maude (1971)"
##  [8,] "Butch Cassidy and the Sundance Kid (1969)"
##  [9,] "Rear Window (1954)"
## [10,] "Magnificent Seven, The (1954)"
##       3
##  [1,] "Star Wars (1977)"
```

```
##  [2,] "Pulp Fiction (1994)"
##  [3,] "Fargo (1996)"
##  [4,] "Willy Wonka and the Chocolate Factory (1971)"
##  [5,] "Godfather, The (1972)"
##  [6,] "Silence of the Lambs, The (1991)"
##  [7,] "Raiders of the Lost Ark (1981)"
##  [8,] "Empire Strikes Back, The (1980)"
##  [9,] "Shawshank Redemption, The (1994)"
## [10,] "Usual Suspects, The (1995)"
##       10
##  [1,] "Schindler's List (1993)"
##  [2,] "Godfather: Part II, The (1974)"
##  [3,] "To Kill a Mockingbird (1962)"
##  [4,] "Blade Runner (1982)"
##  [5,] "Killing Fields, The (1984)"
##  [6,] "Boot, Das (1981)"
##  [7,] "Great Escape, The (1963)"
##  [8,] "Titanic (1997)"
##  [9,] "Princess Bride, The (1987)"
## [10,] "Mr. Smith Goes to Washington (1939)"
##       23
##  [1,] "Usual Suspects, The (1995)"
##  [2,] "Godfather, The (1972)"
##  [3,] "Monty Python and the Holy Grail (1974)"
##  [4,] "Wrong Trousers, The (1993)"
##  [5,] "Schindler's List (1993)"
##  [6,] "Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1963)"
##  [7,] "Shawshank Redemption, The (1994)"
##  [8,] "GoodFellas (1990)"
##  [9,] "2001: A Space Odyssey (1968)"
## [10,] "Trainspotting (1996)"
##       28
##  [1,] "Return of the Jedi (1983)"
##  [2,] "Empire Strikes Back, The (1980)"
##  [3,] "L.A. Confidential (1997)"
##  [4,] "Die Hard (1988)"
##  [5,] "Shawshank Redemption, The (1994)"
##  [6,] "Alien (1979)"
##  [7,] "Reservoir Dogs (1992)"
##  [8,] "Trainspotting (1996)"
##  [9,] "2001: A Space Odyssey (1968)"
## [10,] "Godfather, The (1972)"
```

Using the built in functions from RecommenderLab to evaluate the models,Using K fold validation it can be seen that UBCF performs better than IBCF.

```
#Model Evaluation
#A 5-fold cross validation
n_fold=5
evaluation_scheme <- evaluationScheme(ratings_movies, method="cross-validation", n_fold, given=3, goodRa
evaluation_results <- evaluate(evaluation_scheme, method="UBCF")
```

```
## UBCF run fold/sample [model time/prediction time]
```

```
##   1  [0sec/0.19sec]
##   2  [0.02sec/0.36sec]
##   3  [0sec/0.17sec]
##   4  [0sec/0.19sec]
##   5  [0sec/0.17sec]
##   6  [0.01sec/0.17sec]
##   7  [0.01sec/0.16sec]
##   8  [0.01sec/0.18sec]
##   9  [0sec/0.18sec]
##   10  [0sec/0.17sec]
```

```r
eval_results <- getConfusionMatrix(evaluation_results)[[1]]
eval_results #The evaluation results of the top recommendation
```

```
##           TP        FP       FN       TN precision     recall        TPR
## 1  0.4285714 0.4642857 21.28571 306.8214 0.4800000 0.01975175 0.01975175
## 2  0.6964286 1.0892857 21.01786 306.1964 0.3900000 0.04616786 0.04616786
## 3  0.9821429 1.6964286 20.73214 305.5893 0.3666667 0.05799936 0.05799936
## 4  1.3214286 2.2500000 20.39286 305.0357 0.3700000 0.08164914 0.08164914
## 5  1.5535714 2.9107143 20.16071 304.3750 0.3480000 0.09601582 0.09601582
## 6  1.8392857 3.5178571 19.87500 303.7679 0.3433333 0.11354673 0.11354673
## 7  2.0535714 4.1964286 19.66071 303.0893 0.3285714 0.12593144 0.12593144
## 8  2.2500000 4.8928571 19.46429 302.3929 0.3150000 0.13635315 0.13635315
## 9  2.4464286 5.5892857 19.26786 301.6964 0.3044444 0.14430525 0.14430525
## 10 2.6071429 6.3214286 19.10714 300.9643 0.2920000 0.15177030 0.15177030
##            FPR
## 1  0.001467240
## 2  0.003491493
## 3  0.005445243
## 4  0.007239243
## 5  0.009369848
## 6  0.011319746
## 7  0.013521865
## 8  0.015785044
## 9  0.018024711
## 10 0.020405129
```

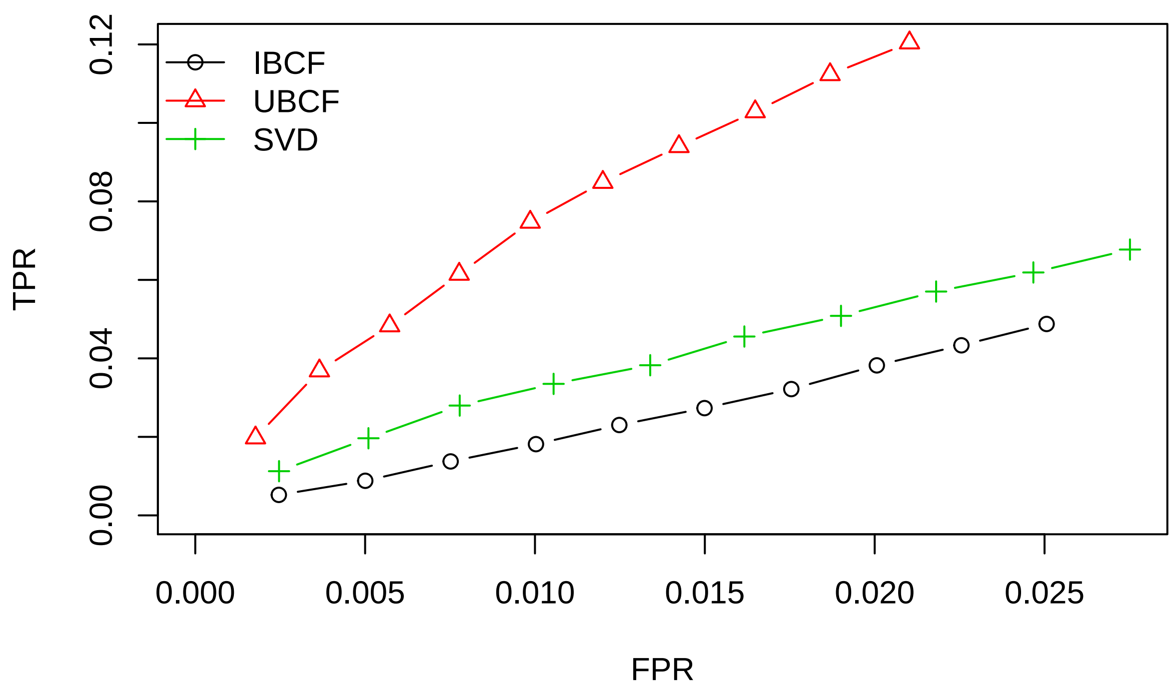
```r
algorithms <- list(
  IBCF = list(name = "IBCF", param = NULL),
  UBCF = list(name = "UBCF", param = NULL),
  SVD = list(name = "SVD", param = NULL)
)

evlist <- evaluate(evaluation_scheme, algorithms)
```

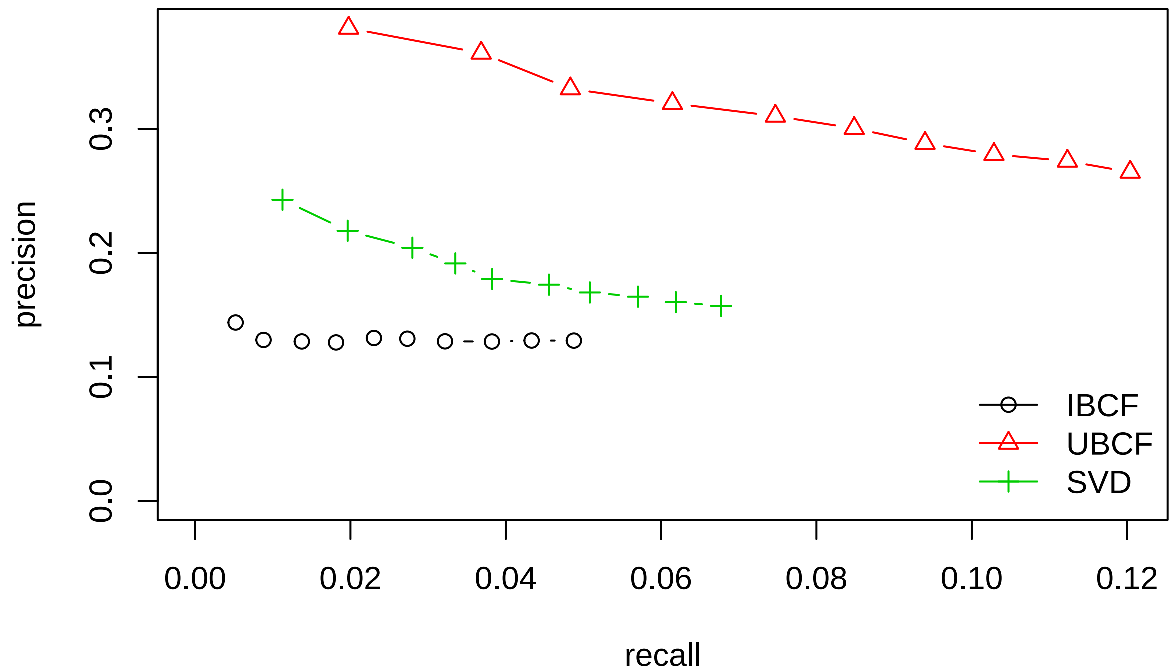
```
## IBCF run fold/sample [model time/prediction time]
##   1  [0.94sec/0.03sec]
##   2  [0.76sec/0.01sec]
##   3  [0.77sec/0.03sec]
##   4  [0.75sec/0.03sec]
##   5  [0.75sec/0.03sec]
##   6  [0.75sec/0.02sec]
##   7  [0.76sec/0.01sec]
```

```
##   8  [0.77sec/0.03sec]
##   9  [0.75sec/0.03sec]
##   10  [0.78sec/0.02sec]
## UBCF run fold/sample [model time/prediction time]
##   1  [0.02sec/0.17sec]
##   2  [0.01sec/0.17sec]
##   3  [0.01sec/0.18sec]
##   4  [0.02sec/0.16sec]
##   5  [0.01sec/0.18sec]
##   6  [0sec/0.18sec]
##   7  [0.01sec/0.19sec]
##   8  [0.02sec/0.17sec]
##   9  [0sec/0.17sec]
##   10  [0sec/0.17sec]
## SVD run fold/sample [model time/prediction time]
##   1  [0.34sec/0.05sec]
##   2  [0.15sec/0.05sec]
##   3  [0.12sec/0.05sec]
##   4  [0.34sec/0.05sec]
##   5  [0.14sec/0.06sec]
##   6  [0.16sec/0.05sec]
##   7  [0.31sec/0.05sec]
##   8  [0.19sec/0.05sec]
##   9  [0.18sec/0.04sec]
##   10  [0.16sec/0.05sec]
```

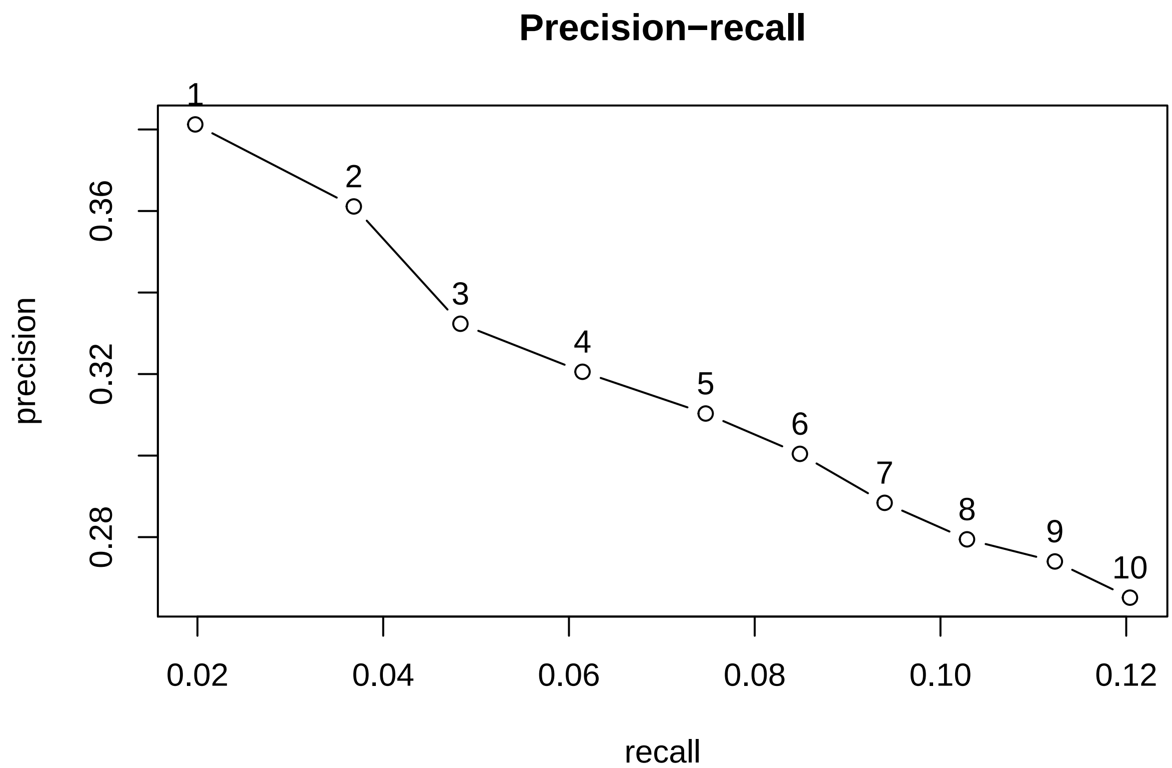
```
plot(evlist, legend="topleft")
```




```
plot(evlist,"prec", legend="bottomright")
```




```
plot(evaluation_results,"prec/rec", annotate = TRUE, main = "Precision-recall")
```



Hierarchial Cluster by Genres

The data has to be re-organize so that it allows for the movies to be displayed by specific genres, this is much easier compared to selecting a movie from a large list of all available movies

Take a subset from the data and calculate the distance between genres,using the Euclidean method which measures the distance between each pair of points within each cluster .After calculating the distance I used a dendogram to display the data with average,complete and ward methods.The distance matrix shows the distance between clusters e.g. the distance between cluster 3 and 20 is 1.73 so this indicates that there are possible good similarity of movies between these two clusters.The larger the distance the more likely that genres are disimilar.

Initially each genre is trated as a single cluster and then the algorithm tries to find the closest in distance until all the genres form one large cluster

```
set.seed(1306)
sammovies <- movies[sample(2:20, 11),] # create sample from the data set

dstncs<- dist(sammovies[2:20], method="euclidean") #The distance between two points

print(dstncs,digits=3)#euclidean distance between each variable
```

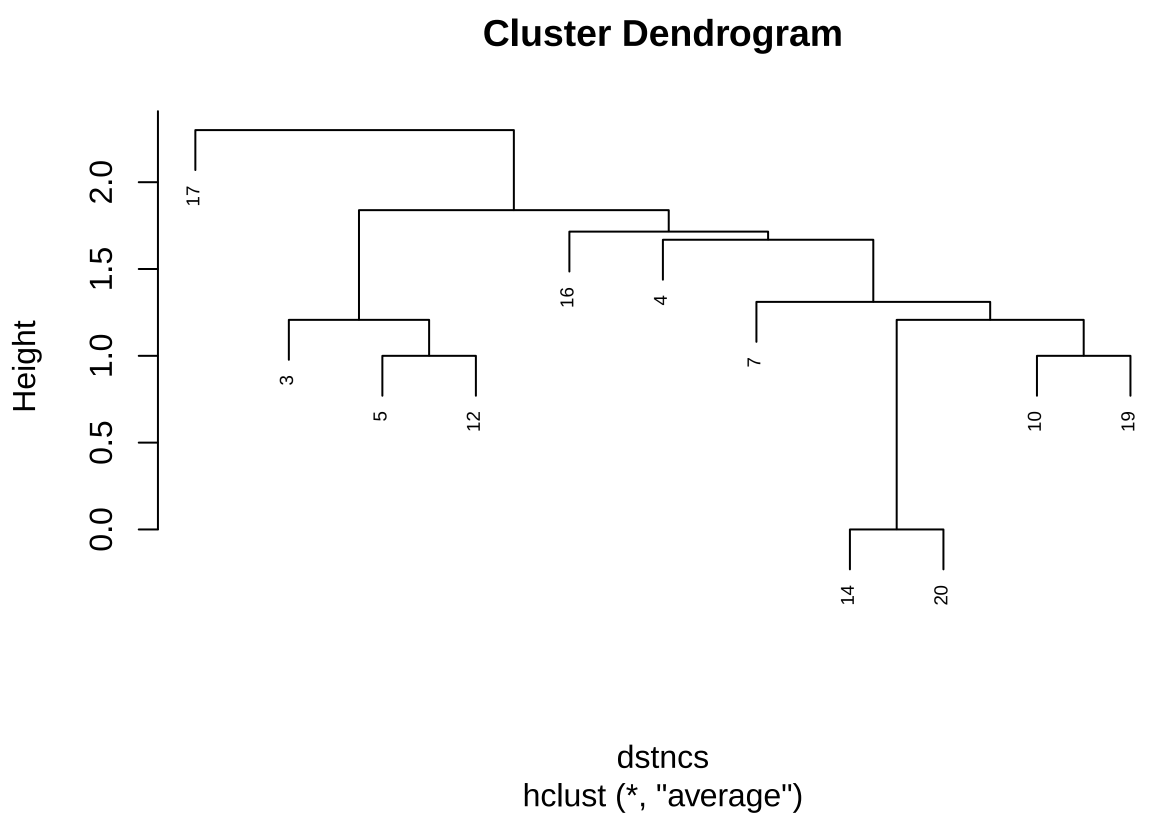
```
##       16    5   10    3   12   14    7   20   17    4
## 5  2.24
## 10 2.00 1.73
## 3  1.73 1.41 1.73
```

```
## 12 2.00 1.00 2.00 1.00
## 14 1.41 1.73 1.41 1.73 2.00
## 7  2.00 1.73 1.41 1.73 2.00 1.41
## 20 1.41 1.73 1.41 1.73 2.00 0.00 1.41
## 17 2.24 2.00 2.65 2.00 1.73 2.65 2.65 2.65
## 4  1.73 2.00 1.73 2.00 2.24 1.73 1.73 1.73 2.00
## 19 1.73 1.41 1.00 1.41 1.73 1.00 1.00 1.00 2.45 1.41
```

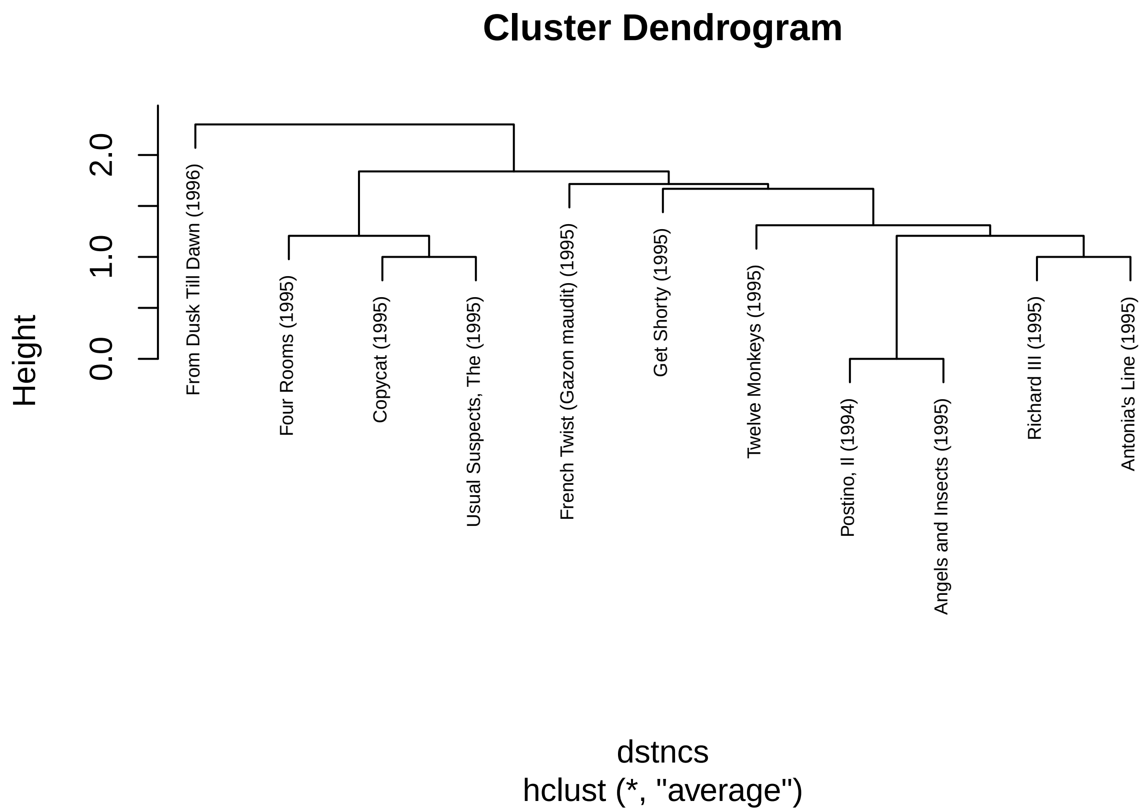
```r
cluster<- hclust(dstncs)#dendogram

cluster<- hclust(dstncs, method="average")#dendogram with average

#The vertical lines represents the distance between clusters
plot(cluster, cex = 0.6)
```




```r
plot(cluster, labels=sammovies$Title,cex = 0.6)
```



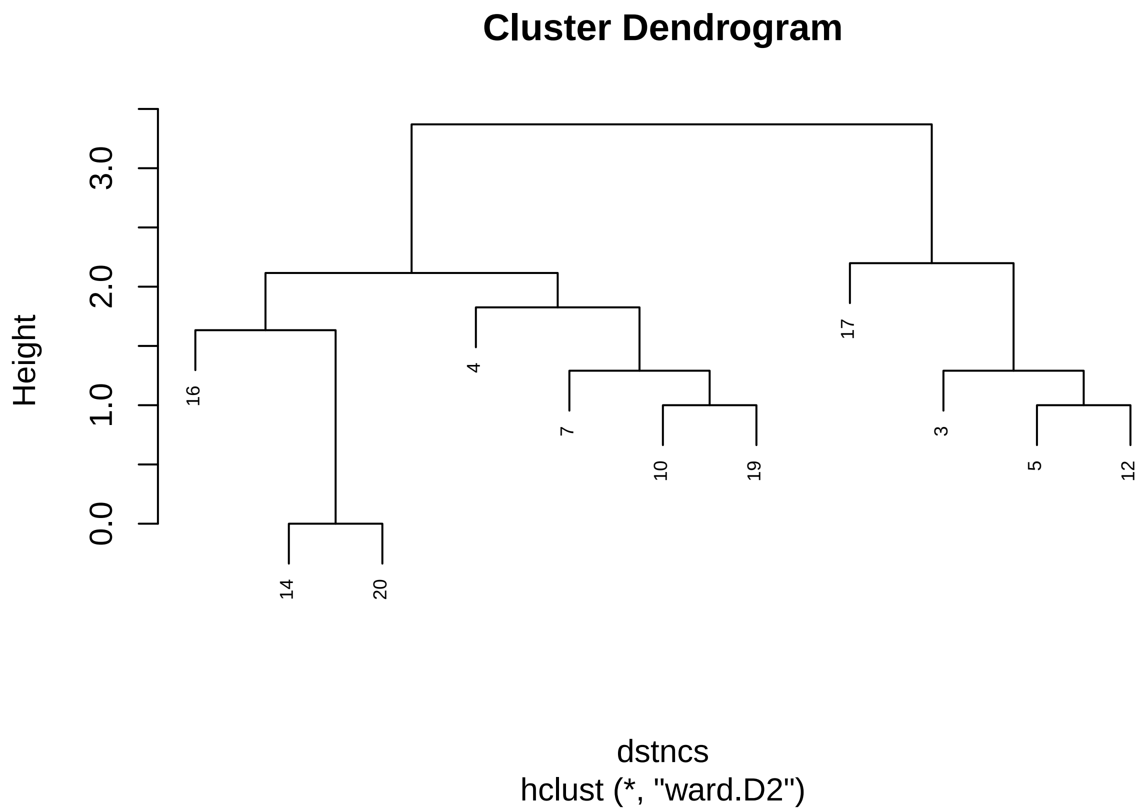
Displaying cluster with Ward.D2 method taking a look at the movies Toy Story and Psycho to see which other movies are similar to them and the genres that they belong to

The Euclidean distance between points p and q is the length of the line segment connecting them

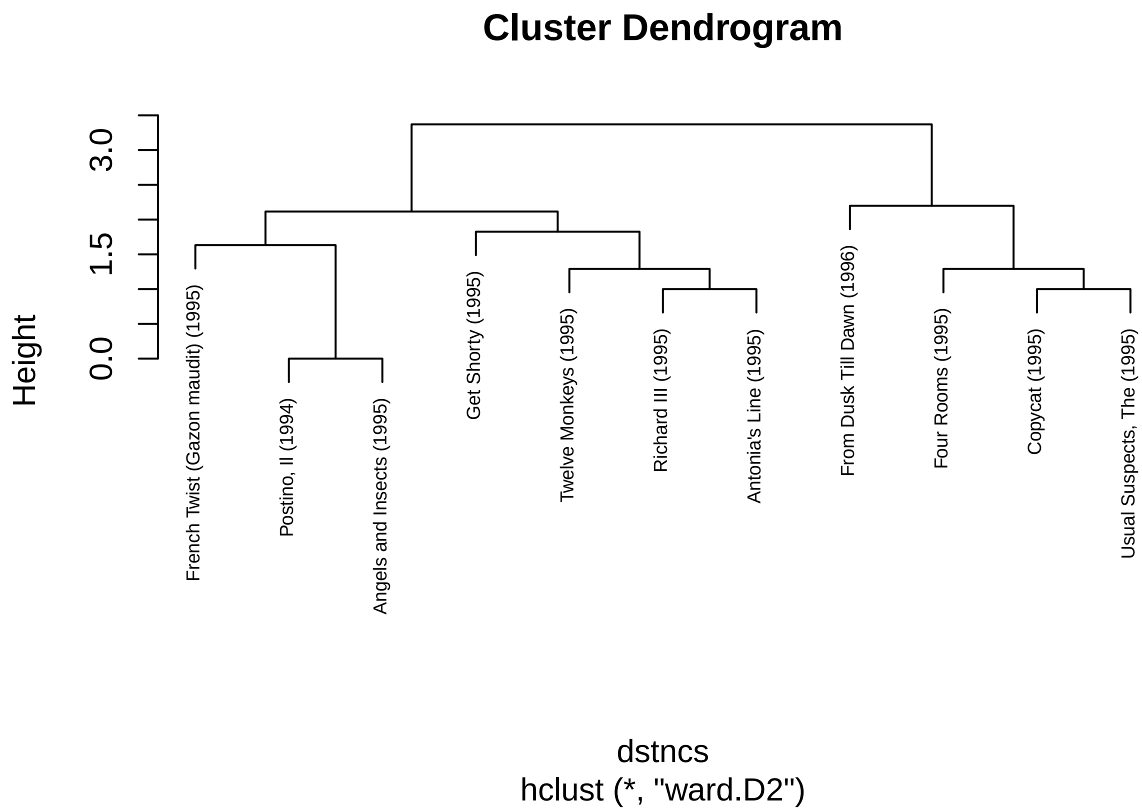
```
clustMovies = hclust(dstncs, method = "ward.D2")
cm=hclust(dstncs, method = "ward.D2")

plot(cm, cex = 0.6)

plot(clustMovies,cex = 0.6)
```




```
plot(clustMovies, labels=sammovies$Title,cex = 0.6)
```




```r
#cuts the dendogram into 10 groups
clustGrps1=cutree(clustMovies, k=10)

# Calculate distances between genre features
distances = dist(movies[2:20], method = "euclidean")

clustMovies = hclust(distances, method = "ward.D2")

#plot(clustMovies, hang=-1,cex = 0.6)

#Label each movie in the clusters
clustGrps = cutree(clustMovies, k=10)
```

```r
#compute the average value and percentage of movies in each genre and for each cluster
tapply(movies$Comedy, clustGrps, mean)
```

```
##          1          2          3          4          5          6
## 0.34645669 0.12790698 0.11111111 0.12195122 0.00000000 0.22033898
##          7          8          9         10
## 1.00000000 0.09166667 0.83809524 0.10280374
```

```r
tapply(movies$Horror, clustGrps, mean)
```

```
##          1          2          3          4          5          6
```

```
## 0.00000000 0.06201550 0.07407407 0.02439024 0.00000000 0.00000000
##          7          8          9         10
## 0.00000000 0.01666667 0.00000000 0.54205607
```

```r
subset(movies, Title=="Toy Story (1995)")# get the cluster that Toy story and Psycho belongs to
```

```
##              Title Unknown Action Adventure Animation Childrens Comedy
## 1 Toy Story (1995)       0      0         0         1         1      1
##   Crime Documentary Drama Fantasy FilmNoir Horror Musical Mystery Romance
## 1     0           0     0       0        0      0       0       0       0
##   SciFi Thriller War Western
## 1     0        0   0       0
```

```r
cluster2 = subset(movies, clustGrps==1)
cluster2$Title[1:10]
```

```
##  [1] Toy Story (1995)
##  [2] Babe (1995)
##  [3] Free Willy 2: The Adventure Home (1995)
##  [4] Santa Clause, The (1994)
##  [5] Lion King, The (1994)
##  [6] Mask, The (1994)
##  [7] Free Willy (1993)
##  [8] Home Alone (1990)
##  [9] Aladdin (1992)
## [10] Snow White and the Seven Dwarfs (1937)
## 1664 Levels: 'Til There Was You (1997) ... Zeus and Roxanne (1997)
```

```r
subset(movies, Title=="Psycho (1960)")
```

```
##             Title Unknown Action Adventure Animation Childrens Comedy
## 185 Psycho (1960)       0      0         0         0         0      0
##     Crime Documentary Drama Fantasy FilmNoir Horror Musical Mystery
## 185     0           0     0       0        0      1       0       0
##     Romance SciFi Thriller War Western
## 185       1     0        1   0       0
```

```r
cluster3 = subset(movies, clustGrps==3)
cluster3$Title[1:10]
```

```
##  [1] Four Rooms (1995)
##  [2] Taxi Driver (1976)
##  [3] Disclosure (1994)
##  [4] Dolores Claiborne (1994)
##  [5] Firm, The (1993)
##  [6] Blade Runner (1982)
##  [7] So I Married an Axe Murderer (1993)
##  [8] Silence of the Lambs, The (1991)
##  [9] Diabolique (1996)
## [10] Lone Star (1996)
## 1664 Levels: 'Til There Was You (1997) ... Zeus and Roxanne (1997)
```

Recommending movies with similar genres allows for a possible wider choice in selection as opposed to UBCF where the recommendation is based on other users rating.Hierarchialclustering usually starts with each variable in its own cluster and then combines the next closest cluster by euclidean method until their is one large cluster.

weighting of variable within the cluster

```
#calculate the cluster means
binsp = split(movies[2:20], clustGrps)
lapply(binsp, colMeans)
```

```
## $`1`
##     Unknown      Action   Adventure   Animation   Childrens      Comedy
## 0.000000000 0.070866142 0.338582677 0.307086614 0.889763780 0.346456693
##       Crime Documentary       Drama     Fantasy    FilmNoir      Horror
## 0.007874016 0.000000000 0.157480315 0.165354331 0.000000000 0.000000000
##     Musical     Mystery     Romance       SciFi    Thriller         War
## 0.149606299 0.000000000 0.031496063 0.062992126 0.007874016 0.000000000
##     Western
## 0.000000000
##
## $`2`
##     Unknown      Action   Adventure   Animation   Childrens      Comedy
## 0.000000000 0.732558140 0.333333333 0.011627907 0.011627907 0.127906977
##       Crime Documentary       Drama     Fantasy    FilmNoir      Horror
## 0.031007752 0.000000000 0.147286822 0.000000000 0.000000000 0.062015504
##     Musical     Mystery     Romance       SciFi    Thriller         War
## 0.000000000 0.007751938 0.046511628 0.317829457 0.298449612 0.038759690
##     Western
## 0.100775194
##
## $`3`
##     Unknown      Action   Adventure   Animation   Childrens      Comedy
##  0.01234568  0.03703704  0.00000000  0.00000000  0.01234568  0.11111111
##       Crime Documentary       Drama     Fantasy    FilmNoir      Horror
##  0.09876543  0.00000000  0.31481481  0.00617284  0.14197531  0.07407407
##     Musical     Mystery     Romance       SciFi    Thriller         War
##  0.00000000  0.35185185  0.06790123  0.03703704  0.79012346  0.00000000
##     Western
##  0.00000000
##
## $`4`
##     Unknown      Action   Adventure   Animation   Childrens      Comedy
##  0.00000000  0.17073171  0.01219512  0.00000000  0.00000000  0.12195122
##       Crime Documentary       Drama     Fantasy    FilmNoir      Horror
##  0.97560976  0.00000000  0.48780488  0.00000000  0.00000000  0.02439024
##     Musical     Mystery     Romance       SciFi    Thriller         War
##  0.00000000  0.01219512  0.07317073  0.01219512  0.32926829  0.00000000
##     Western
##  0.00000000
##
## $`5`
##     Unknown      Action   Adventure   Animation   Childrens      Comedy
##           0           0           0           0           0           0
```

```
##       Crime Documentary       Drama     Fantasy    FilmNoir      Horror 
##           0           0           1           0           0           0 
##     Musical     Mystery     Romance       SciFi    Thriller         War 
##           0           0           0           0           0           0 
##     Western 
##           0 
## 
## $`6`
##     Unknown      Action   Adventure   Animation   Childrens      Comedy 
##  0.00000000  0.22033898  0.03389831  0.00000000  0.00000000  0.22033898 
##       Crime Documentary       Drama     Fantasy    FilmNoir      Horror 
##  0.00000000  0.01694915  0.59322034  0.00000000  0.00000000  0.00000000 
##     Musical     Mystery     Romance       SciFi    Thriller         War 
##  0.00000000  0.00000000  0.20338983  0.03389831  0.05084746  1.00000000 
##     Western 
##  0.01694915 
## 
## $`7`
##     Unknown      Action   Adventure   Animation   Childrens      Comedy 
##   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   1.0000000 
##       Crime Documentary       Drama     Fantasy    FilmNoir      Horror 
##   0.0000000   0.0000000   0.2372263   0.0000000   0.0000000   0.0000000 
##     Musical     Mystery     Romance       SciFi    Thriller         War 
##   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000   0.0000000 
##     Western 
##   0.0000000 
## 
## $`8`
##     Unknown      Action   Adventure   Animation   Childrens      Comedy 
## 0.000000000 0.116666667 0.000000000 0.000000000 0.000000000 0.091666667 
##       Crime Documentary       Drama     Fantasy    FilmNoir      Horror 
## 0.016666667 0.000000000 0.666666667 0.000000000 0.008333333 0.016666667 
##     Musical     Mystery     Romance       SciFi    Thriller         War 
## 0.000000000 0.000000000 1.000000000 0.000000000 0.091666667 0.008333333 
##     Western 
## 0.000000000 
## 
## $`9`
##     Unknown      Action   Adventure   Animation   Childrens      Comedy 
##  0.00000000  0.03809524  0.00952381  0.00000000  0.01904762  0.83809524 
##       Crime Documentary       Drama     Fantasy    FilmNoir      Horror 
##  0.00000000  0.00000000  0.08571429  0.00000000  0.00000000  0.00000000 
##     Musical     Mystery     Romance       SciFi    Thriller         War 
##  0.35238095  0.00000000  0.75238095  0.00952381  0.00952381  0.00952381 
##     Western 
##  0.00000000 
## 
## $`10`
##     Unknown      Action   Adventure   Animation   Childrens      Comedy 
##  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.10280374 
##       Crime Documentary       Drama     Fantasy    FilmNoir      Horror 
##  0.00000000  0.45794393  0.07476636  0.00000000  0.00000000  0.54205607 
##     Musical     Mystery     Romance       SciFi    Thriller         War 
##  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000  0.00000000 
```

```
##     Western
##  0.00000000
```