SDM2_HW2

Sri Pavankrishna Yenugu

2024-03-08

Q1.(Modified Exercise 14.4 in ESL) Cluster the demographic data (>data(marketing in ESL package)) of Table 14.1 using "generalized association rules" with a classification tree. (This data can also be found on the ESL website). Specifically, generate a reference sample the same size as the training set, by either (a) randomly permuting the columns independently, or by (b) sampling from a uniform distribution the values within each feature. Build a classification tree to the training sample (class 1) and the reference sample (class 0) and describe the terminal nodes having highest estimated class 1 probability.

```r
library(ISLR2)
library(arules)
```

```
## Loading required package: Matrix

##
## Attaching package: 'arules'

## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```r
#install.packages("gtools")
library(gtools)
library(rpart)
library(rpart.plot)
mark_data <- read.table("/Users/sripavanyenugu/Downloads/marketing.data.txt")
str(mark_data)
```

```
## 'data.frame':    8993 obs. of  14 variables:
##  $ V1 : int  9 9 9 1 1 8 1 6 2 4 ...
##  $ V2 : int  2 1 2 2 2 1 1 1 1 1 ...
##  $ V3 : int  1 1 1 5 5 1 5 3 1 1 ...
##  $ V4 : int  5 5 3 1 1 6 2 3 6 7 ...
##  $ V5 : int  4 5 5 2 2 4 3 4 3 4 ...
##  $ V6 : int  5 5 1 6 6 8 9 3 8 8 ...
##  $ V7 : int  5 5 5 5 3 5 4 5 5 4 ...
##  $ V8 : int  3 3 2 1 1 3 1 1 3 3 ...
##  $ V9 : int  3 5 3 4 4 2 3 1 3 2 ...
##  $ V10: int  0 2 1 2 2 0 1 0 0 0 ...
##  $ V11: int  1 1 2 3 3 1 2 2 2 2 ...
##  $ V12: int  1 1 3 1 1 1 3 3 3 3 ...
##  $ V13: int  7 7 7 7 7 7 7 7 7 7 ...
##  $ V14: int  NA 1 1 1 1 1 1 1 1 1 ...
```

```
head(mark_data)
```

```
##   V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14
## 1  9  2  1  5  4  5  5  3  3   0   1   1   7  NA
## 2  9  1  1  5  5  5  5  3  5   2   1   1   7   1
## 3  9  2  1  3  5  1  5  2  3   1   2   3   7   1
## 4  1  2  5  1  2  6  5  1  4   2   3   1   7   1
## 5  1  2  5  1  2  6  3  1  4   2   3   1   7   1
## 6  8  1  1  6  4  8  5  3  2   0   1   1   7   1
```

```
market <- na.omit(mark_data)
rownames(market) <- NULL
attr(market, "na.action") <- NULL
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:arules':
##
##     intersect, recode, setdiff, setequal, union

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

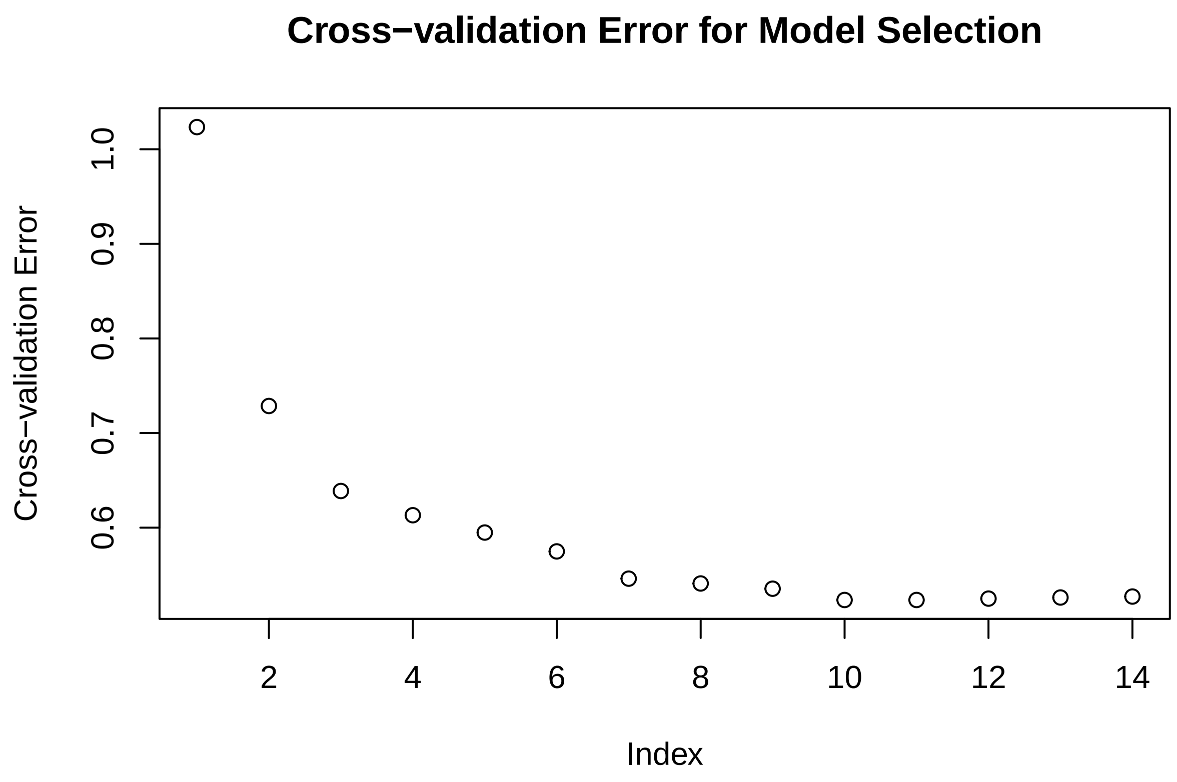
```
marke <- mutate_if(market, is.numeric, as.factor)

set.seed(123)
samp <- lapply(marke, function(x){sample(x, replace=TRUE) })
samp <- as.data.frame(samp)
marke$class <- 1
samp$class <- 0

data <- rbind(marke, samp)
data1 = data[sample(nrow(data)),]
```

#Classification Tree

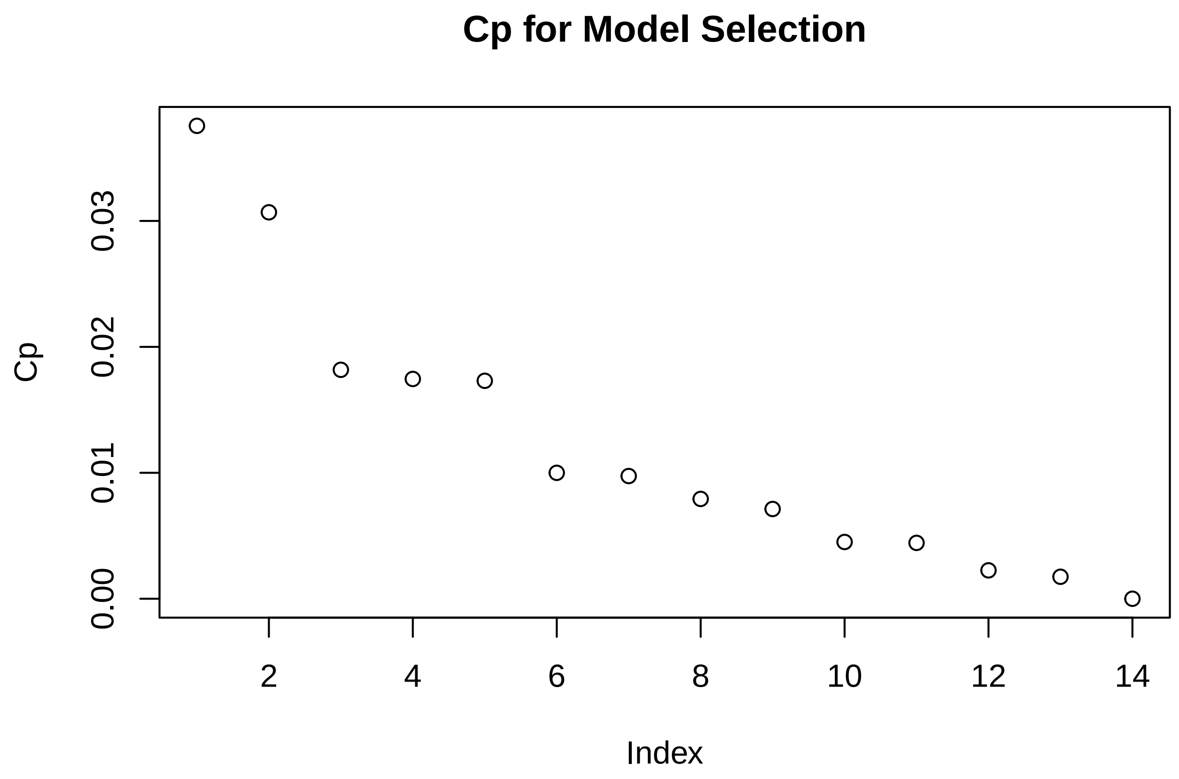
```
library("rpart")
set.seed(123)
model.control <- rpart.control(minsplit = 600, cp = 0)
fit_W <- rpart(class~., data = data1, method = "class", control =
model.control)
names(fit_W)
```

```
##  [1] "frame"               "where"               "call"
##  [4] "terms"               "cptable"             "method"
##  [7] "parms"               "control"             "functions"
## [10] "numresp"             "splits"              "csplit"
## [13] "variable.importance" "y"                   "ordered"
```

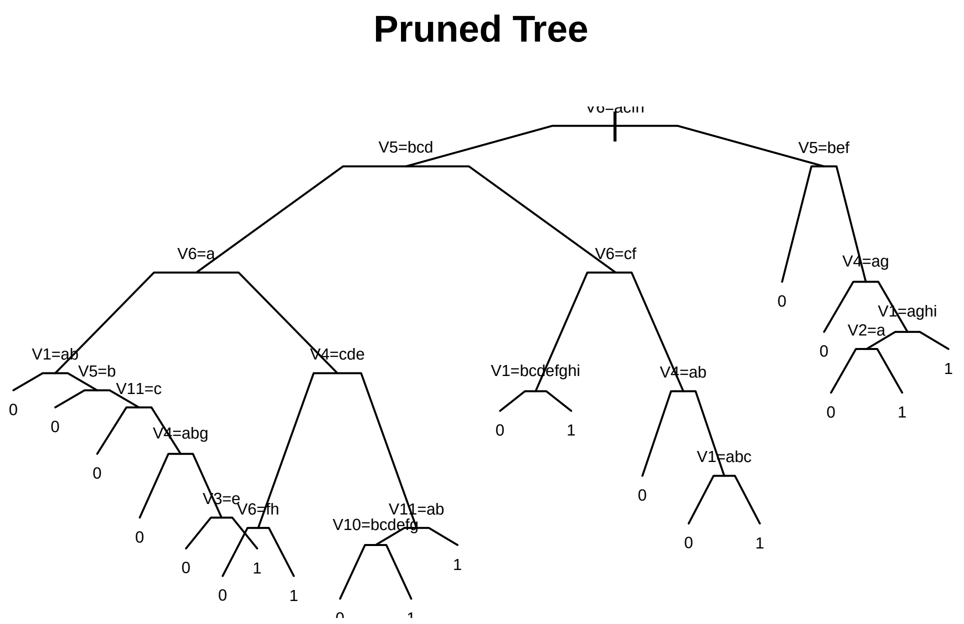
```
plot(fit_W$cptable[,4], main = "Cross-validation Error for Model Selection", ylab = "Cross-validation E
```



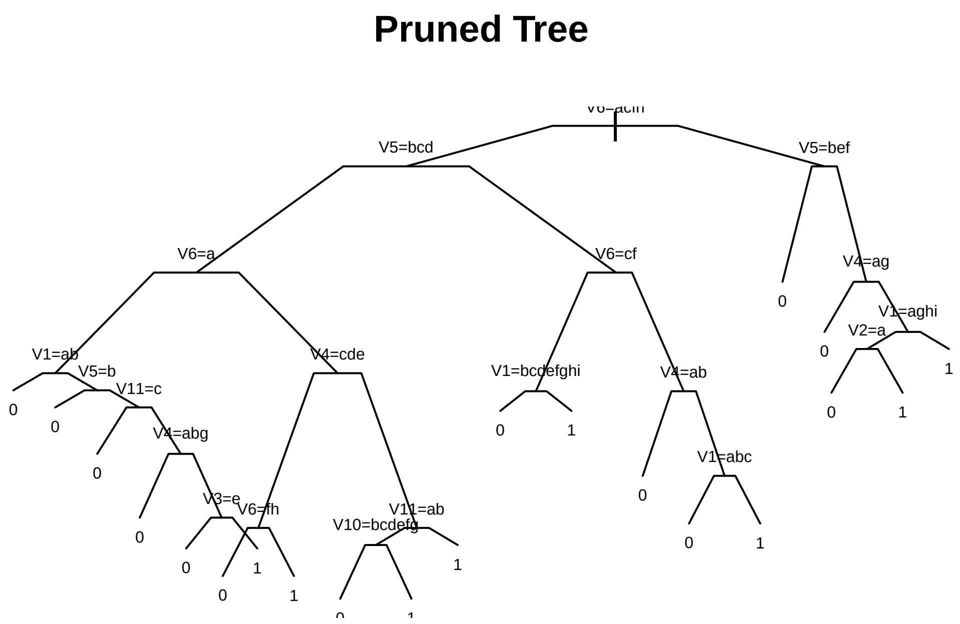
```
plot(fit_W$cptable[,1], main = "Cp for Model Selection", ylab = "Cp")
```



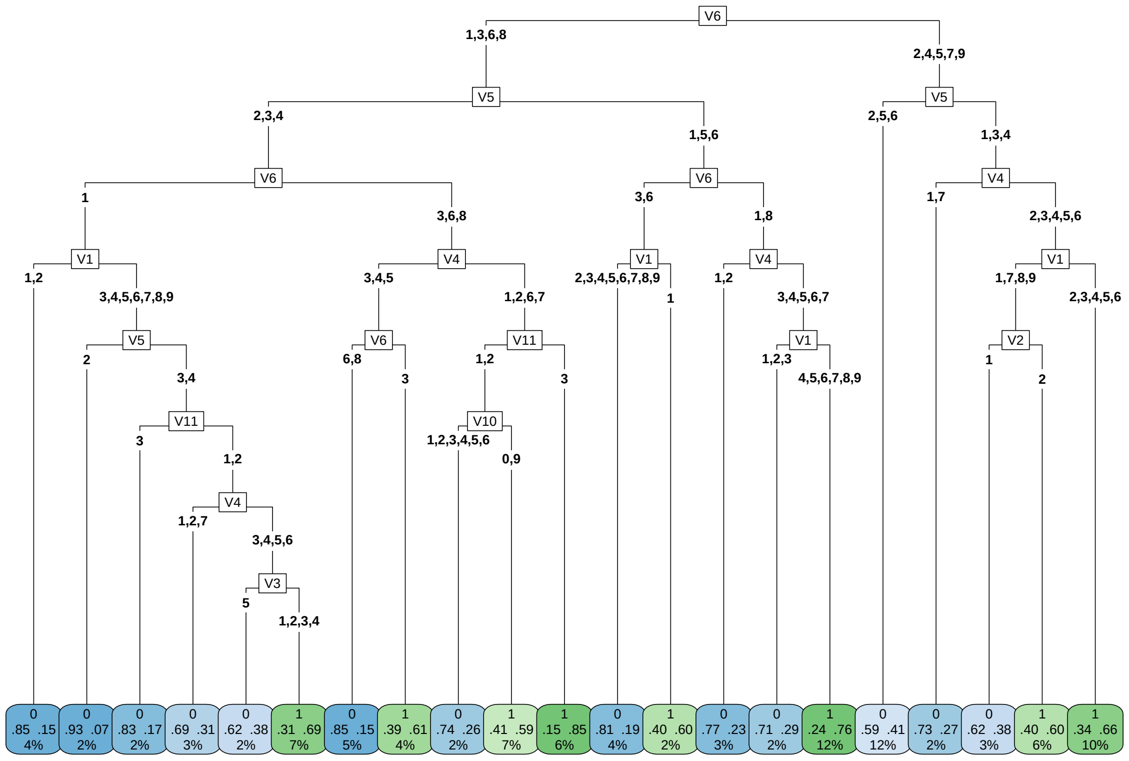
```r
min_cp_index <- which.min(fit_W$cptable[, 4])

pruned_fit_W <- prune(fit_W, cp = fit_W$cptable[min_cp_index, 1])

plot(pruned_fit_W, branch = 0.3, compress = TRUE, main = "Pruned Tree")
text(pruned_fit_W, cex = 0.5)
```



```r
plot(pruned_fit_W, branch = .3, compress = T, main = "Pruned Tree")
text(pruned_fit_W, cex = .5)
```



```r
library(rpart.plot)
rpart.plot(pruned_fit_W, type = 5, extra = 104, cex = 0.4)
```



```r
set.seed(123)
y_pred <- predict(pruned_fit_W, data1, type = 'class')
misclassification_rate <- mean(y_pred != data1$class)
misclassification_rate
```

```
## [1] 0.295957
```

Conclusion:

After pruning the tree, 4 major terminal nodes for class-1 were found, each with good observations and high class-1 percentages:

- Nodes: 7% (69% 1s), 6% (85% 1s), 12% (78% 1s), 10% (66% 1s).
- 54% of data is in class-1 nodes, close to 50%, with a misclassification rate of 18.78%.
- Key variables for class-1 separation: ANNUAL INCOME OF HOUSEHOLD, HOUSEHOLDER STATUS, OCCUPATION, EDUCATION, AGE.

SDM2_HW2

Sri Pavankrishna Yenugu

2024-03-08

2) Consider the California Housing Data from KAGGLE. You are going to apply association rules to this data. (https://www.kaggle.com/camnugent/california- housing-prices).

a. Read in the data and transfer to a binary incidence matrix. Visualize this matrix.

```r
setwd("/Users/sripavanyenugu/Downloads")

library(arules)
```

```
## Loading required package: Matrix

##
## Attaching package: 'arules'

## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```r
# Read the data
my_housing_data <- read.csv("housing.csv")

my_housing_data$ocean_proximity <- as.factor(my_housing_data$ocean_proximity)

my_numeric_cols <- c("housing_median_age", "total_rooms", "total_bedrooms",
                  "population", "households", "median_income",
                  "latitude", "longitude")

my_housing_data[, my_numeric_cols] <- lapply(my_housing_data[, my_numeric_cols], as.factor)
my_housing_data$median_house_value <- cut(my_housing_data$median_house_value, breaks = 5)

# Convert to binary matrix
my_housing_trans <- as(my_housing_data, "transactions")
my_housing_rules <- apriori(my_housing_trans, parameter = list(support = 0.001, confidence = 0.5))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.5    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 20
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[28248 item(s), 20640 transaction(s)] done [0.03s].
## sorting and recoding items ... [1422 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.01s].
## writing ... [1574 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```r
inspect(head(my_housing_rules))
```

```
##     lhs                     rhs                          support     confidence
## [1] {latitude=36.3}      => {ocean_proximity=INLAND}    0.001017442 1.0000000
## [2] {longitude=-121.41} => {ocean_proximity=INLAND}    0.001017442 1.0000000
## [3] {longitude=-121.01} => {ocean_proximity=INLAND}    0.001017442 1.0000000
## [4] {latitude=36.73}     => {ocean_proximity=INLAND}    0.001017442 1.0000000
## [5] {longitude=-117.39} => {ocean_proximity=INLAND}    0.001017442 0.9545455
## [6] {longitude=-118.58} => {ocean_proximity=<1H OCEAN} 0.001065891 1.0000000
##     coverage    lift     count
## [1] 0.001017442 3.150664 21
## [2] 0.001017442 3.150664 21
## [3] 0.001017442 3.150664 21
## [4] 0.001017442 3.150664 21
## [5] 0.001065891 3.007452 21
## [6] 0.001065891 2.259194 22
```

b. What are the top three high lift rules?

```r
my_trans <- as(my_housing_data, "transactions")
my_rules <- apriori(my_trans, parameter = list(support = 0.01, confidence = 0.5))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.5    0.1    1 none FALSE            TRUE       5    0.01      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 206
##
```

```
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[28248 item(s), 20640 transaction(s)] done [0.03s].
## sorting and recoding items ... [55 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [17 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```r
top_three_lift_rules <- inspect(sort(my_rules, by = "lift", decreasing = TRUE)[1:3])
```

```
##     lhs                                            rhs                                             support 
## [1] {median_house_value=(1.45e+04,1.12e+05]} => {ocean_proximity=INLAND}                  0.16632752
## [2] {ocean_proximity=INLAND}                    => {median_house_value=(1.45e+04,1.12e+05]} 0.16632752
## [3] {housing_median_age=14}                     => {ocean_proximity=INLAND}                  0.01022287
```

c. What are the top 4 rules according to confidence?

```r
inspect(sort(my_rules,by="confidence",decreasing=TRUE)[1:4])
```

```
##     lhs                                            rhs                          support confidence
## [1] {median_house_value=(1.45e+04,1.12e+05]} => {ocean_proximity=INLAND}    0.16632752  0.7647583 0.
## [2] {housing_median_age=35,
##      median_house_value=(1.12e+05,2.09e+05]} => {ocean_proximity=<1H OCEAN} 0.01133721  0.6704871 0.
## [3] {housing_median_age=36,
##      median_house_value=(1.12e+05,2.09e+05]} => {ocean_proximity=<1H OCEAN} 0.01090116  0.6465517 0.
## [4] {median_house_value=(2.09e+05,3.06e+05]} => {ocean_proximity=<1H OCEAN} 0.13425388  0.6066112 0.
```

d. A person comes to you and wants to purchase an average priced home as close to the ocean as possible. What can you recommend and/or what can the expect? Use association rules to guide you.

Recommendations: -For an average-priced home between $14,500 to $112,000, expect properties INLAND. -For homes priced between $112,000 to $209,000 and with median ages 35 or 36, expect properties near the ocean. These recommendations are based on association rules from the dataset.

e. What characteristics in the data associate with low population areas?

INLAND Locations: -Lower median incomes $0.5 to $1.75 are associated with INLAND locations. -Lower median house values $14,500 to $112,000 are also associated with INLAND areas. -Younger median housing ages (around 28) are linked to INLAND areas.

Low Population Areas: -Lower median incomes, lower house values, and younger housing ages are characteristics associated with low population areas in the dataset. -These associations suggest that INLAND locations, with their lower median incomes and house values, tend to have lower population densities.

SDM2_HW2

Sri Pavankrishna Yenugu

2024-03-08

3.Consider the MovieLense data that is available in the recommenderlab package >data(MovieLense) >?MovieLense The data was collected through the MovieLens web site during a seven- month, and contains about 100,000 ratings (1-5) from 943 users on 1664 movies. See the help file on the data to understand how to best manipulate the object. a) Develop a user-based recommender system. Create the system so that outputs a user's top ten recommendations. Demo it on five users.

```r
#install.packages("recommenderlab")
library(recommenderlab)
```

```
## Loading required package: Matrix

## Loading required package: arules

##
## Attaching package: 'arules'

## The following objects are masked from 'package:base':
##
##     abbreviate, write

## Loading required package: proxy

##
## Attaching package: 'proxy'

## The following object is masked from 'package:Matrix':
##
##     as.matrix

## The following objects are masked from 'package:stats':
##
##     as.dist, dist

## The following object is masked from 'package:base':
##
##     as.matrix

## Registered S3 methods overwritten by 'registry':
##   method               from
##   print.registry_field proxy
##   print.registry_entry proxy
```

```r
data(MovieLense)
head(MovieLense)
```

```
## 6 x 1664 rating matrix of class 'realRatingMatrix' with 789 ratings.
```

```r
rec_model <- Recommender(data = MovieLense, method = "UBCF")
usr_ids <- c(1, 100, 200, 300, 400)
i <- 1
while (i <= length(usr_ids)) {
  usr_id <- usr_ids[i]
  usr_recs <- predict(rec_model, MovieLense[usr_id, ], n = 10)
  movie_ids <- as(usr_recs, "list")[[1]]
  cat("User:", usr_id, "\n")
  print(movie_ids)
  cat("\n")
  i <- i + 1
}
```

```
## User: 1
##  [1] "Boot, Das (1981)"
##  [2] "Matilda (1996)"
##  [3] "Winter Guest, The (1997)"
##  [4] "She's the One (1996)"
##  [5] "Manchurian Candidate, The (1962)"
##  [6] "City of Lost Children, The (1995)"
##  [7] "Double vie de Veronique, La (Double Life of Veronique, The) (1991)"
##  [8] "187 (1997)"
##  [9] "Leaving Las Vegas (1995)"
## [10] "Wag the Dog (1997)"
##
## User: 100
##  [1] "Angels and Insects (1995)"
##  [2] "Muriel's Wedding (1994)"
##  [3] "Jungle Book, The (1994)"
##  [4] "Innocents, The (1961)"
##  [5] "Delta of Venus (1994)"
##  [6] "Everyone Says I Love You (1996)"
##  [7] "Down by Law (1986)"
##  [8] "C'est arrive pres de chez vous (1992)"
##  [9] "Manhattan (1979)"
## [10] "Breaking the Waves (1996)"
##
## User: 200
##  [1] "French Kiss (1995)"
##  [2] "Gabbeh (1996)"
##  [3] "Trainspotting (1996)"
##  [4] "Before Sunrise (1995)"
##  [5] "Anastasia (1997)"
##  [6] "Much Ado About Nothing (1993)"
##  [7] "Die Hard (1988)"
##  [8] "Pillow Book, The (1995)"
##  [9] "Mrs. Brown (Her Majesty, Mrs. Brown) (1997)"
## [10] "To Kill a Mockingbird (1962)"
```

```
##
## User: 300
##  [1] "Snow White and the Seven Dwarfs (1937)"
##  [2] "Houseguest (1994)"
##  [3] "Alice in Wonderland (1951)"
##  [4] "Another Stakeout (1993)"
##  [5] "Bullets Over Broadway (1994)"
##  [6] "Pollyanna (1960)"
##  [7] "How to Make an American Quilt (1995)"
##  [8] "Friday (1995)"
##  [9] "In the Name of the Father (1993)"
## [10] "Beverly Hills Cop III (1994)"
##
## User: 400
##  [1] "Fear of a Black Hat (1993)"
##  [2] "Serial Mom (1994)"
##  [3] "Brady Bunch Movie, The (1995)"
##  [4] "Heavy Metal (1981)"
##  [5] "Multiplicity (1996)"
##  [6] "C'est arrive pres de chez vous (1992)"
##  [7] "Kolya (1996)"
##  [8] "Mirror Has Two Faces, The (1996)"
##  [9] "Matilda (1996)"
## [10] "Craft, The (1996)"
```

b) For the same users in part A, predict their top 10 recommendations based on an item-based recommender system. How do they compare?

```r
library(recommenderlab)
data(MovieLense)
usr_ids <- c(1, 100, 200, 300, 400)
rec_model_ibcf <- Recommender(data = MovieLense, method = "IBCF")
i <- 1
while (i <= length(usr_ids)) {
  usr_id <- usr_ids[i]
  usr_recs <- predict(rec_model_ibcf, MovieLense[usr_id, ], n = 10)
  movie_ids <- as(usr_recs, "list")[[1]]
  cat("User:", usr_id, " (IBCF)\n")
  print(movie_ids)
  cat("\n")
  i <- i + 1
}
```

```
## User: 1  (IBCF)
## [1] "Entertaining Angels: The Dorothy Day Story (1996)"
## [2] "Big Bang Theory, The (1994)"
## [3] "They Made Me a Criminal (1939)"
## [4] "Cyclo (1995)"
## [5] "King of New York (1990)"
## [6] "Very Natural Thing, A (1974)"
## [7] "Walk in the Sun, A (1945)"
## [8] "Hush (1998)"
##
```

```
## User: 100  (IBCF)
##  [1] "I Don’t Want to Talk About It (De eso no se habla) (1993)"
##  [2] "Wife, The (1995)"
##  [3] "Little City (1998)"
##  [4] "Mamma Roma (1962)"
##  [5] "Shopping (1994)"
##  [6] "Nemesis 2: Nebula (1995)"
##  [7] "Bewegte Mann, Der (1994)"
##  [8] "Entertaining Angels: The Dorothy Day Story (1996)"
##  [9] "Star Kid (1997)"
## [10] "King of New York (1990)"
##
## User: 200  (IBCF)
##  [1] "Mad Love (1995)"                        "D3: The Mighty Ducks (1996)"
##  [3] "Turbo: A Power Rangers Movie (1997)" "Pillow Book, The (1995)"
##  [5] "Air Bud (1997)"                         "Deceiver (1997)"
##  [7] "Incognito (1997)"                       "Bio-Dome (1996)"
##  [9] "Striking Distance (1993)"               "Beverly Hills Ninja (1997)"
##
## User: 300  (IBCF)
##  [1] "Sunchaser, The (1996)"            "War at Home, The (1996)"
##  [3] "Mat’ i syn (1997)"                "B. Monkey (1998)"
##  [5] "You So Crazy (1994)"              "Fear, The (1995)"
##  [7] "Time Tracers (1995)"              "Prefontaine (1997)"
##  [9] "Other Voices, Other Rooms (1997)" "New York Cop (1996)"
##
## User: 400  (IBCF)
##  [1] "Power 98 (1995)"
##  [2] "Bloody Child, The (1996)"
##  [3] "Paris Was a Woman (1995)"
##  [4] "They Made Me a Criminal (1939)"
##  [5] "Cyclo (1995)"
##  [6] "New York Cop (1996)"
##  [7] "Very Natural Thing, A (1974)"
##  [8] "Walk in the Sun, A (1945)"
##  [9] "Three Lives and Only One Death (1996)"
## [10] "Butterfly Kiss (1995)"
```

You can see that there are significant differences between the two sets of recomendations for each user by comparing them. This is due to the fact that different fundamental principles are used by filtering algorithms that are item based and user based. While item based filtering proposess items comparable to those the user has already liked, user based filtering suggests goods enjoyed by similar users.

c) Consider the first user part A. Are the goals of a recommender system met – relevance, novelty, serendipity, diversity? Comment on each of the best you can given the information you have.

User 100 generally likes Drama, Comedy, and Thriller kind of movies.

1. Relevance:

- It refers to how well the recommended items match the user’s preferences or interests. From the list of top 10 recommendations for User 1, we see movie titles such as “Boot, Das (1981)”, “Matilda (1996)”, “Winter Guest, The (1997)”, “Leaving Las Vegas (1995)”, and “Wag the Dog (1997)”. These recommendations seem relevant as they align with the user's previously rated movies like "Matilda (1996)" and "Leaving Las Vegas (1995)". Therefore, in terms of relevance, the recommender system seems to be meeting the goal for User 1.

2. Novelty:

- It refers to recommending items that the user has not seen or experienced before. In this case, we do not have direct information on whether the user has seen the recommended movies before. However, we can assume that movies like "Boot, Das (1981)", "Winter Guest, The (1997)", and "Double vie de Veronique, La (Double Life of Veronique, The) (1991)" might offer some novelty to the user, especially if they have not watched these movies previously. So, the recommender system appears to provide some level of novelty.

3. Serendipity:

- It refers to surprising the user with unexpected recommendations that they end up liking. It's hard to gauge serendipity directly from the list provided, but movies like "Manchurian Candidate, The (1962)" or "City of Lost Children, The (1995)" might offer serendipitous recommendations if they are outside the user's typical genre preferences. However, without more information on the user's viewing habits, it's challenging to assess the level of serendipity.

4.Diversity: - It refers to providing a range of items from various genres or categories. Looking at the list, we see a mix of genres such as drama, comedy, and thriller. This mix indicates that the recommender system is trying to provide a diverse set of recommendations to User 1. However, without knowing more about the user's specific preferences, it's difficult to determine if this level of diversity is sufficient.

In conclusion, based on the provided list of recommendations for User 1, the recommender system appears to be meeting the goals of relevance and possibly novelty. The level of serendipity and diversity is harder to assess without more detailed information about the user's preferences and viewing history.