**Problem 1 (20 points). Consider the following transactional database.**

(1) Mine all frequent itemsets using the Apriori algorithm that we discussed in the class. Show all candidate itemsets and frequent itemsets. You should follow the step by step process that we discussed in the class (i.e., C1 → L1 → C2 → L2 → …). You don't need to show the pruning steps. Minimum support = 50% (or 3 or more transactions). To save your time, L1 is given below:

Frequent 1-itemsets:

items support count

[1] {5} 0.8333333 5

[2] {3} 0.8333333 5

[3] {4} 0.6666667 4

[4] {2} 0.6666667 4

[5] {1} 0.6666667 4

[6] {6} 0.5000000 3

[7] {8} 0.5000000 3

Frequent 2-itemsets:

items support count

[1] {5, 6} 0.5000000 3

[2] {3, 6} 0.5000000 3

[3] {2, 6} 0.5000000 3

[4] {1, 5} 0.5000000 3

[5] {1, 3} 0.5000000 3

[6] {2, 5} 0.6666667 4

[7] {2, 3} 0.6666667 4

[8] {2, 4} 0.5000000 3

[9] {4, 5} 0.6666667 4

[10] {3, 4} 0.5000000 3

[11] {3, 5} 0.6666667 4

[12] {5} 0.8333333 5

[13] {3} 0.8333333 5

[14] {4} 0.6666667 4

[15] {2} 0.6666667 4

[16] {1} 0.6666667 4

[17] {6} 0.5000000 3

[18] {8} 0.5000000 3

Frequent 3-itemsets:

items support count

[1] {2, 5, 6} 0.5000000 3

[2] {2, 3, 6} 0.5000000 3

[3] {3, 5, 6} 0.5000000 3

[4] {5, 6} 0.5000000 3

[5] {3, 6} 0.5000000 3

[6] {2, 6} 0.5000000 3

[7] {1, 5} 0.5000000 3

[8] {1, 3} 0.5000000 3

[9] {2, 4, 5} 0.5000000 3

[10] {2, 3, 4} 0.5000000 3

[11] {2, 3, 5} 0.6666667 4

[12] {2, 5} 0.6666667 4

[13] {2, 3} 0.6666667 4

[14] {2, 4} 0.5000000 3

[15] {3, 4, 5} 0.5000000 3

[16] {4, 5} 0.6666667 4

[17] {3, 4} 0.5000000 3

[18] {3, 5} 0.6666667 4

[19] {5} 0.8333333 5

[20] {3} 0.8333333 5

[21] {4} 0.6666667 4

[22] {2} 0.6666667 4

[23] {1} 0.6666667 4

[24] {6} 0.5000000 3

[25] {8} 0.5000000 3

Frequent 4-itemsets:

items support count

[1] {2, 3, 5, 6} 0.5000000 3

[2] {2, 5, 6} 0.5000000 3

[3] {2, 3, 6} 0.5000000 3

[4] {3, 5, 6} 0.5000000 3

[5] {5, 6} 0.5000000 3

[6] {3, 6} 0.5000000 3

[7] {2, 6} 0.5000000 3

[8] {1, 5} 0.5000000 3

[9] {1, 3} 0.5000000 3

[10] {2, 3, 4, 5} 0.5000000 3

[11] {2, 4, 5} 0.5000000 3

[12] {2, 3, 4} 0.5000000 3

[13] {2, 3, 5} 0.6666667 4

[14] {2, 5} 0.6666667 4

[15] {2, 3} 0.6666667 4

[16] {2, 4} 0.5000000 3

[17] {3, 4, 5} 0.5000000 3

[18] {4, 5} 0.6666667 4

[19] {3, 4} 0.5000000 3

[20] {3, 5} 0.6666667 4

[21] {5} 0.8333333 5

[22] {3} 0.8333333 5

[23] {4} 0.6666667 4

[24] {2} 0.6666667 4

[25] {1} 0.6666667 4

[26] {6} 0.5000000 3

[27] {8} 0.5000000 3

Frequent 5-itemsets:

items support count

[1] {2, 3, 5, 6} 0.5000000 3

[2] {2, 5, 6} 0.5000000 3

[3] {2, 3, 6} 0.5000000 3

[4] {3, 5, 6} 0.5000000 3

[5] {5, 6} 0.5000000 3

[6] {3, 6} 0.5000000 3

[7] {2, 6} 0.5000000 3

[8] {1, 5} 0.5000000 3

[9] {1, 3} 0.5000000 3

[10] {2, 3, 4, 5} 0.5000000 3

[11] {2, 4, 5} 0.5000000 3

[12] {2, 3, 4} 0.5000000 3

[13] {2, 3, 5} 0.6666667 4

[14] {2, 5} 0.6666667 4

[15] {2, 3} 0.6666667 4

[16] {2, 4} 0.5000000 3

[17] {3, 4, 5} 0.5000000 3

[18] {4, 5} 0.6666667 4

[19] {3, 4} 0.5000000 3

[20] {3, 5} 0.6666667 4

[21] {5} 0.8333333 5

[22] {3} 0.8333333 5

[23] {4} 0.6666667 4

[24] {2} 0.6666667 4

[25] {1} 0.6666667 4

[26] {6} 0.5000000 3

[27] {8} 0.5000000 3

Frequent 6-itemsets:

items support count

[1] {2, 3, 5, 6} 0.5000000 3

[2] {2, 5, 6} 0.5000000 3

[3] {2, 3, 6} 0.5000000 3

[4] {3, 5, 6} 0.5000000 3

[5] {5, 6} 0.5000000 3

[6] {3, 6} 0.5000000 3

[7] {2, 6} 0.5000000 3

[8] {1, 5} 0.5000000 3

[9] {1, 3} 0.5000000 3

[10] {2, 3, 4, 5} 0.5000000 3

[11] {2, 4, 5} 0.5000000 3

[12] {2, 3, 4} 0.5000000 3

[13] {2, 3, 5} 0.6666667 4

[14] {2, 5} 0.6666667 4

[15] {2, 3} 0.6666667 4

[16] {2, 4} 0.5000000 3

[17] {3, 4, 5} 0.5000000 3

[18] {4, 5} 0.6666667 4

[19] {3, 4} 0.5000000 3

[20] {3, 5} 0.6666667 4

[21] {5} 0.8333333 5

[22] {3} 0.8333333 5

[23] {4} 0.6666667 4

[24] {2} 0.6666667 4

[25] {1} 0.6666667 4

[26] {6} 0.5000000 3

[27] {8} 0.5000000 3

Frequent 7-itemsets:

items support count

[1] {2, 3, 5, 6} 0.5000000 3

[2] {2, 5, 6} 0.5000000 3

[3] {2, 3, 6} 0.5000000 3

[4] {3, 5, 6} 0.5000000 3

[5] {5, 6} 0.5000000 3

[6] {3, 6} 0.5000000 3

[7] {2, 6} 0.5000000 3

[8] {1, 5} 0.5000000 3

[9] {1, 3} 0.5000000 3

[10] {2, 3, 4, 5} 0.5000000 3

[11] {2, 4, 5} 0.5000000 3

[12] {2, 3, 4} 0.5000000 3

[13] {2, 3, 5} 0.6666667 4

[14] {2, 5} 0.6666667 4

[15] {2, 3} 0.6666667 4

[16] {2, 4} 0.5000000 3

[17] {3, 4, 5} 0.5000000 3

[18] {4, 5} 0.6666667 4

[19] {3, 4} 0.5000000 3

[20] {3, 5} 0.6666667 4

[21] {5} 0.8333333 5

[22] {3} 0.8333333 5

[23] {4} 0.6666667 4

[24] {2} 0.6666667 4

[25] {1} 0.6666667 4

[26] {6} 0.5000000 3

[27] {8} 0.5000000 3

Frequent 8-itemsets:

items support count

[1] {2, 3, 5, 6} 0.5000000 3

[2] {2, 5, 6} 0.5000000 3

[3] {2, 3, 6} 0.5000000 3

[4] {3, 5, 6} 0.5000000 3

[5] {5, 6} 0.5000000 3

[6] {3, 6} 0.5000000 3

[7] {2, 6} 0.5000000 3

[8] {1, 5} 0.5000000 3

[9] {1, 3} 0.5000000 3

[10] {2, 3, 4, 5} 0.5000000 3

[11] {2, 4, 5} 0.5000000 3

[12] {2, 3, 4} 0.5000000 3

[13] {2, 3, 5} 0.6666667 4

[14] {2, 5} 0.6666667 4

[15] {2, 3} 0.6666667 4

[16] {2, 4} 0.5000000 3

[17] {3, 4, 5} 0.5000000 3

[18] {4, 5} 0.6666667 4

[19] {3, 4} 0.5000000 3

[20] {3, 5} 0.6666667 4

[21] {5} 0.8333333 5

[22] {3} 0.8333333 5

[23] {4} 0.6666667 4

[24] {2} 0.6666667 4

[25] {1} 0.6666667 4

[26] {6} 0.5000000 3

[27] {8} 0.5000000 3

Frequent 9-itemsets:

items support count

[1] {2, 3, 5, 6} 0.5000000 3

[2] {2, 5, 6} 0.5000000 3

[3] {2, 3, 6} 0.5000000 3

[4] {3, 5, 6} 0.5000000 3

[5] {5, 6} 0.5000000 3

[6] {3, 6} 0.5000000 3

[7] {2, 6} 0.5000000 3

[8] {1, 5} 0.5000000 3

[9] {1, 3} 0.5000000 3

[10] {2, 3, 4, 5} 0.5000000 3

[11] {2, 4, 5} 0.5000000 3

[12] {2, 3, 4} 0.5000000 3

[13] {2, 3, 5} 0.6666667 4

[14] {2, 5} 0.6666667 4

[15] {2, 3} 0.6666667 4

[16] {2, 4} 0.5000000 3

[17] {3, 4, 5} 0.5000000 3

[18] {4, 5} 0.6666667 4

[19] {3, 4} 0.5000000 3

[20] {3, 5} 0.6666667 4

[21] {5} 0.8333333 5

[22] {3} 0.8333333 5

[23] {4} 0.6666667 4

[24] {2} 0.6666667 4

[25] {1} 0.6666667 4

[26] {6} 0.5000000 3

[27] {8} 0.5000000 3

Frequent 10-itemsets:

items support count

[1] {2, 3, 5, 6} 0.5000000 3

[2] {2, 5, 6} 0.5000000 3

[3] {2, 3, 6} 0.5000000 3

[4] {3, 5, 6} 0.5000000 3

[5] {5, 6} 0.5000000 3

[6] {3, 6} 0.5000000 3

[7] {2, 6} 0.5000000 3

[8] {1, 5} 0.5000000 3

[9] {1, 3} 0.5000000 3

[10] {2, 3, 4, 5} 0.5000000 3

[11] {2, 4, 5} 0.5000000 3

[12] {2, 3, 4} 0.5000000 3

[13] {2, 3, 5} 0.6666667 4

[14] {2, 5} 0.6666667 4

[15] {2, 3} 0.6666667 4

[16] {2, 4} 0.5000000 3

[17] {3, 4, 5} 0.5000000 3

[18] {4, 5} 0.6666667 4

[19] {3, 4} 0.5000000 3

[20] {3, 5} 0.6666667 4

[21] {5} 0.8333333 5

[22] {3} 0.8333333 5

[23] {4} 0.6666667 4

[24] {2} 0.6666667 4

[25] {1} 0.6666667 4

[26] {6} 0.5000000 3

[27] {8} 0.5000000 3

(2) Sort all frequent 4-itemsets by their item number. Then, select the first frequent 4-

itemset from the sorted list of frequent 4-itemsets and mine all strong rules from this

itemset that have the format {W, X} => {Y, Z}, where W, X, Y, and Z are individual

items. Assume that minimum confidence = 80%.

> inspect(rules)

lhs rhs support confidence coverage lift count

[1] {} => {3} 0.8333333 0.8333333 1.0000000 1.00 5

[2] {} => {5} 0.8333333 0.8333333 1.0000000 1.00 5

[3] {6} => {2} 0.5000000 1.0000000 0.5000000 1.50 3

[4] {6} => {3} 0.5000000 1.0000000 0.5000000 1.20 3

[5] {6} => {5} 0.5000000 1.0000000 0.5000000 1.20 3

[6] {4} => {5} 0.6666667 1.0000000 0.6666667 1.20 4

[7] {5} => {4} 0.6666667 0.8000000 0.8333333 1.20 4

[8] {2} => {3} 0.6666667 1.0000000 0.6666667 1.20 4

[9] {3} => {2} 0.6666667 0.8000000 0.8333333 1.20 4

[10] {2} => {5} 0.6666667 1.0000000 0.6666667 1.20 4

[11] {5} => {2} 0.6666667 0.8000000 0.8333333 1.20 4

[12] {3} => {5} 0.6666667 0.8000000 0.8333333 0.96 4

[13] {5} => {3} 0.6666667 0.8000000 0.8333333 0.96 4

[14] {2, 6} => {3} 0.5000000 1.0000000 0.5000000 1.20 3

[15] {3, 6} => {2} 0.5000000 1.0000000 0.5000000 1.50 3

[16] {2, 6} => {5} 0.5000000 1.0000000 0.5000000 1.20 3

[17] {5, 6} => {2} 0.5000000 1.0000000 0.5000000 1.50 3

[18] {3, 6} => {5} 0.5000000 1.0000000 0.5000000 1.20 3

[19] {5, 6} => {3} 0.5000000 1.0000000 0.5000000 1.20 3

[20] {2, 4} => {3} 0.5000000 1.0000000 0.5000000 1.20 3

[21] {3, 4} => {2} 0.5000000 1.0000000 0.5000000 1.50 3

[22] {2, 4} => {5} 0.5000000 1.0000000 0.5000000 1.20 3

[23] {3, 4} => {5} 0.5000000 1.0000000 0.5000000 1.20 3

[24] {2, 3} => {5} 0.6666667 1.0000000 0.6666667 1.20 4

[25] {2, 5} => {3} 0.6666667 1.0000000 0.6666667 1.20 4

[26] {3, 5} => {2} 0.6666667 1.0000000 0.6666667 1.50 4

[27] {2, 3, 6} => {5} 0.5000000 1.0000000 0.5000000 1.20 3

[28] {2, 5, 6} => {3} 0.5000000 1.0000000 0.5000000 1.20 3

[29] {3, 5, 6} => {2} 0.5000000 1.0000000 0.5000000 1.50 3

[30] {2, 3, 4} => {5} 0.5000000 1.0000000 0.5000000 1.20 3

[31] {2, 4, 5} => {3} 0.5000000 1.0000000 0.5000000 1.20 3

[32] {3, 4, 5} => {2} 0.5000000 1.0000000 0.5000000 1.50 3

**Code:**library(arules)

transactions <- list(

c("2", "3", "4", "5", "6", "8"),

c("1", "2", "3", "5", "6"),

c("1", "4", "5", "7", "8"),

c("2", "3", "4", "5", "6"),

c("1", "2", "3", "4", "5", "7"),

c("1", "3", "8")

)

trans <- as(transactions, "transactions")

min\_support <- 0.5

max\_len <- 10

frequent\_itemsets\_list <- list()

# Generate frequent itemsets of each length through looping

for (len in 1:max\_len) {

frequent\_itemsets <- eclat(trans, parameter = list(support = min\_support, maxlen = len))

# If the number of new frequent itemsets is 0, exit the loop

if (length(frequent\_itemsets) == 0) {

break

}

frequent\_itemsets\_list[[len]] <- frequent\_itemsets

}

# Print all frequent itemsets

for (len in 1:length(frequent\_itemsets\_list)) {

cat(paste0("\nFrequent ", len, "-itemsets:\n"))

inspect(frequent\_itemsets\_list[[len]])

}

# Find all association rules with a confidence greater than or equal to 0.8

rules <- apriori(trans, parameter = list(support = min\_support, confidence = 0.8))

inspect(rules)

**Problem 2 (20 points). Consider the following contingency table.**

(1). Compute the lift, all-confidence, cosine, Kulczynski and imbalance ratio measure, and determine whether buying coffee and buying tea are positively correlated, negatively correlated, or not correlated.

Lift: 1.201632

All-confidence: 0.5954357

Cosine: 0.7174462

Kulczynski: 0.7299468

Imbalance Ratio: -0.001100665

(2). Perform the chi-square test with 5% significance level and determine whether they are correlated or not.

Chi-squared: 67.18003

P-value: 2.478121e-16

Given the p-value is less than 0.05, we reject the null hypothesis and conclude that buying coffee and buying tea are correlated.

**Code:**  
# Contingency table data

matrix\_data <- matrix(c(287, 45, 195, 143), ncol=2, byrow=TRUE)

rownames(matrix\_data) <- c("Tea = Yes", "Tea = No")

colnames(matrix\_data) <- c("Coffee = Yes", "Coffee = No")

# Calculate association measures

a <- matrix\_data[1,1]

b <- matrix\_data[1,2]

c <- matrix\_data[2,1]

d <- matrix\_data[2,2]

n <- sum(matrix\_data)

lift <- (a \* n) / ((a + c) \* (a + b))

all\_confidence <- min(a / (a + b), a / (a + c))

cosine <- a / (sqrt((a + b) \* (a + c)))

kulczynski <- 0.5 \* (a / (a + b) + a / (a + c))

imbalance\_ratio <- abs(a / (a + b) - a / (a + c)) / (a / (a + b) + a / (a + c) - 2 \* (a^2) / n)

# Print association measures

cat("Lift:", lift, "\n")

cat("All-confidence:", all\_confidence, "\n")

cat("Cosine:", cosine, "\n")

cat("Kulczynski:", kulczynski, "\n")

cat("Imbalance Ratio:", imbalance\_ratio, "\n")

# Chi-squared test

chisq\_test <- chisq.test(matrix\_data)

cat("Chi-squared:", chisq\_test$statistic, "\n")

cat("P-value:", chisq\_test$p.value, "\n")

if(chisq\_test$p.value < 0.05) {

cat("Given the p-value is less than 0.05, we reject the null hypothesis and conclude that buying coffee and buying tea are correlated.\n")

} else {

cat("Given the p-value is greater than 0.05, we fail to reject the null hypothesis and conclude that buying coffee and buying tea are not correlated.\n")

}

Problem 3 (20 points). This problem is about mining frequent itemsets and strong rules

from a grocery data using R. Use hw7.csv data for this problem. It is a slightly modified

version of the groceries.csv data that is included in R.

(1). Mine all frequent itemsets with the minimum support = 0.005 and include the

screenshot of the summary in your submission. 文本

描述已自动生成

(2). What is the total number of frequent itemsets? List the top five frequent itemsets and their supports in descending order of support.

> cat("\nTotal number of frequent itemsets:", length(frequent\_itemsets), "\n")

Total number of frequent itemsets: 10625253

> cat("Top 5 frequent itemsets based on support:\n")

Top 5 frequent itemsets based on support:

> inspect(head(sort(frequent\_itemsets, by="support"), 5))

items support count

[1] {Column.29=} 0.9996 4998

[2] {Column.28=} 0.9994 4997

[3] {Column.28=, Column.29=} 0.9994 4997

[4] {Column.27=} 0.9992 4996

[5] {Column.27=, Column.28=} 0.9992 4996

(3). What is the total number of frequent 1-itemsets (L1)? List the top five frequent 1-

itemsets and their supports in descending order of support.

> frequent\_1\_itemsets <- subset(frequent\_itemsets, size(items) == 1)

> cat("\nTotal number of frequent 1-itemsets:", length(frequent\_1\_itemsets), "\n")

Total number of frequent 1-itemsets: 227

> cat("Top 5 frequent 1-itemsets based on support:\n")

Top 5 frequent 1-itemsets based on support:

> inspect(head(sort(frequent\_1\_itemsets, by="support"), 5))

items support count

[1] {Column.29=} 0.9996 4998

[2] {Column.28=} 0.9994 4997

[3] {Column.27=} 0.9992 4996

[4] {Column.26=} 0.9990 4995

[5] {Column.25=} 0.9990 4995

(4). What is the total number of frequent 2-itemsets (L2)? List the top five frequent 2-

itemsets and their supports in descending order of support.

> frequent\_2\_itemsets <- subset(frequent\_itemsets, size(items) == 2)

> cat("\nTotal number of frequent 2-itemsets:", length(frequent\_2\_itemsets), "\n")

Total number of frequent 2-itemsets: 4894

> cat("Top 5 frequent 2-itemsets based on support:\n")

Top 5 frequent 2-itemsets based on support:

> inspect(head(sort(frequent\_2\_itemsets, by="support"), 5))

items support count

[1] {Column.28=, Column.29=} 0.9994 4997

[2] {Column.27=, Column.28=} 0.9992 4996

[3] {Column.27=, Column.29=} 0.9992 4996

[4] {Column.25=, Column.26=} 0.9990 4995

[5] {Column.24=, Column.26=} 0.9990 4995

(5). What is the total number of frequent 3-itemsets (L3)? Show the frequent 3-itemset

and its support that has the highest support.

> frequent\_3\_itemsets <- subset(frequent\_itemsets, size(items) == 3)

> cat("\nTotal number of frequent 3-itemsets:", length(frequent\_3\_itemsets), "\n")

Total number of frequent 3-itemsets: 53370

> cat("The frequent 3-itemset with the highest support:\n")

The frequent 3-itemset with the highest support:

> inspect(head(sort(frequent\_3\_itemsets, by="support"), 1))

items support count

[1] {Column.27=, Column.28=, Column.29=} 0.9992 4996

(6). What is the total number of frequent 4-itemsets (L4)? Show the frequent 4-itemset

and its support that has the highest support.

> frequent\_4\_itemsets <- subset(frequent\_itemsets, size(items) == 4)

> cat("\nTotal number of frequent 4-itemsets:", length(frequent\_4\_itemsets), "\n")

Total number of frequent 4-itemsets: 382129

> cat("The frequent 4-itemset with the highest support:\n")

The frequent 4-itemset with the highest support:

> inspect(head(sort(frequent\_4\_itemsets, by="support"), 1))

items support count

[1] {Column.24=, Column.25=, Column.26=, Column.27=} 0.999 4995

(7). Mine all rules with minimum support = 0.005 and minimum confidence = 0.15 and

include the screenshot of the summary in your submission. 文本

描述已自动生成

(8). Sort rules in descending order of confidences and show the top five rules and their

confidences.

cat("\nTop 5 rules based on confidence:\n")

inspect(head(sort(rules, by="confidence"), 5))

(9). Sort rules in descending order of supports and show the top five rules and their

supports.

cat("\nTop 5 rules based on support:\n")

inspect(head(sort(rules, by="support"), 5))

(10). Show all rules that include coffee (in either side of a rule) along with their supports

and confidences.

cat("\nRules that include coffee:\n")

coffee\_rules\_lhs <- subset(rules, subset = lhs %in% "coffee")

coffee\_rules\_rhs <- subset(rules, subset = rhs %in% "coffee")

coffee\_rules <- unique(c(coffee\_rules\_lhs, coffee\_rules\_rhs))

inspect(coffee\_rules)

Code:  
library(arules)

data <- read.csv("hw7.csv", stringsAsFactors = TRUE)

transactions <- as(data, "transactions")

# (1). Mine all frequent itemsets with the minimum support = 0.005

frequent\_itemsets <- apriori(transactions, parameter = list(support = 0.005, target = "frequent itemsets"))

cat("Summary of all frequent itemsets:\n")

summary(frequent\_itemsets)

# (2). Total number of frequent itemsets and top 5 based on support

cat("\nTotal number of frequent itemsets:", length(frequent\_itemsets), "\n")

cat("Top 5 frequent itemsets based on support:\n")

inspect(head(sort(frequent\_itemsets, by="support"), 5))

# (3). Total number of frequent 1-itemsets and top 5

frequent\_1\_itemsets <- subset(frequent\_itemsets, size(items) == 1)

cat("\nTotal number of frequent 1-itemsets:", length(frequent\_1\_itemsets), "\n")

cat("Top 5 frequent 1-itemsets based on support:\n")

inspect(head(sort(frequent\_1\_itemsets, by="support"), 5))

# (4). Total number of frequent 2-itemsets and top 5

frequent\_2\_itemsets <- subset(frequent\_itemsets, size(items) == 2)

cat("\nTotal number of frequent 2-itemsets:", length(frequent\_2\_itemsets), "\n")

cat("Top 5 frequent 2-itemsets based on support:\n")

inspect(head(sort(frequent\_2\_itemsets, by="support"), 5))

# (5). Total number of frequent 3-itemsets and the one with highest support

frequent\_3\_itemsets <- subset(frequent\_itemsets, size(items) == 3)

cat("\nTotal number of frequent 3-itemsets:", length(frequent\_3\_itemsets), "\n")

cat("The frequent 3-itemset with the highest support:\n")

inspect(head(sort(frequent\_3\_itemsets, by="support"), 1))

# (6). Total number of frequent 4-itemsets and the one with highest support

frequent\_4\_itemsets <- subset(frequent\_itemsets, size(items) == 4)

cat("\nTotal number of frequent 4-itemsets:", length(frequent\_4\_itemsets), "\n")

cat("The frequent 4-itemset with the highest support:\n")

inspect(head(sort(frequent\_4\_itemsets, by="support"), 1))

# (7). Mine all rules with minimum support = 0.005 and minimum confidence = 0.15

rules <- apriori(transactions, parameter = list(support = 0.005, confidence = 0.15, target = "rules"))

cat("\nSummary of all rules:\n")

summary(rules)

# (8). Top 5 rules based on confidence

cat("\nTop 5 rules based on confidence:\n")

inspect(head(sort(rules, by="confidence"), 5))

# (9). Top 5 rules based on support

cat("\nTop 5 rules based on support:\n")

inspect(head(sort(rules, by="support"), 5))

# (10). Rules that include coffee

cat("\nRules that include coffee:\n")

coffee\_rules\_lhs <- subset(rules, subset = lhs %in% "coffee")

coffee\_rules\_rhs <- subset(rules, subset = rhs %in% "coffee")

coffee\_rules <- unique(c(coffee\_rules\_lhs, coffee\_rules\_rhs))

inspect(coffee\_rules)