Lab 8: Association Rules --Titanic

You will need the packages arules and arulesViz.

1. Load the dataframe titanic.raw.RData.
   - Save the file to your onedrive
   - load(file.choose()), Select the file. You should see titanic.raw in your Environment window.
   - Check out its structure: str(titanic.raw)

2.Summarize the data in each column.
   - What percent of the records are males? Females?

     Female Male
     21.35393% 78.64607%
   - Give the percentages for the passengers' classes.

     1st 2nd 3rd Crew
     14.76602% 12.94866% 32.07633% 40.20900%
   - Give the percentages for children and adults.

     Adult Child
     95.047706% 4.952294%
   - Check that there are no missing values.

```
> summary(titanic.raw)
  Class         Sex          Age       Survived
 1st :325   Female: 470   Adult:2092   No :1490
 2nd :285   Male  :1731   Child: 109   Yes: 711
 3rd :706
 Crew:885
```

     No Missing values for any of my variables.

2. a. Determine the association rules. You really are only interested in rules with survival status as the consequent. In other words, rhs = c("Survived = No", "Survived = Yes"). Play around with support and confidence levels.
b. Sort your rules by lift and inspect your rules. Provide a copy of your rules. Interpret what you learn from your rules.

```
> inspect(survival_rule_sorted)
    lhs                                    rhs            support    confidence coverage   lift     count
[1] {Class=2nd, Age=Child}              => {Survived=Yes} 0.01090413 1.0000000  0.01090413 3.095640  24
[2] {Class=1st, Sex=Female}             => {Survived=Yes} 0.06406179 0.9724138  0.06587915 3.010243 141
[3] {Class=1st, Sex=Female, Age=Adult} => {Survived=Yes} 0.06360745 0.9722222  0.06542481 3.009650 140
[4] {Class=2nd, Sex=Female}             => {Survived=Yes} 0.04225352 0.8773585  0.04815993 2.715986  93
[5] {Class=2nd, Sex=Female, Age=Adult} => {Survived=Yes} 0.03634711 0.8602151  0.04225352 2.662916  80
[6] {Class=2nd, Sex=Male, Age=Adult}   => {Survived=No}  0.06996820 0.9166667  0.07632894 1.354083 154
[7] {Class=2nd, Sex=Male}              => {Survived=No}  0.06996820 0.8603352  0.08132667 1.270871 154
[8] {Class=3rd, Sex=Male, Age=Adult}   => {Survived=No}  0.17582917 0.8376623  0.20990459 1.237379 387
[9] {Class=3rd, Sex=Male}              => {Survived=No}  0.19173103 0.8274510  0.23171286 1.222295 422
```

From my rules I know that:

1. children in second class who survived account for 1% of my dataset. 100% of these children survived.
2. females in 1st class that survived account for 6.41% of my dataset. 97.24% of them survived
3. Adult females in 1st class that survived account for 6.36% of my dataset. 97.2% of them survived.
4. females in 2nd class that survived account for 4.3% of my dataset. 87.74% of them survived
5. Adult females in 2nd class that survived account for 3.6% of my dataset. 86% of them survived.
6. Adult Males in the 2nd class that did not survive account for 7% of my dataset. 91% of them did not survive.
7. Males in 2nd class that did not survive account for 7% of my dataset. 86% of them did not survive.
8. Adult Males in the 3rd who did not survive account for 17% of my dataset. 83% of them did not survive
9. Males in the 3rd class who did not survive account for 19.17% of my dataset. 82.75% of them did not survive.

3. Some of the rules listed in 2 may be redundant. If so, you might try this. Otherwise, skip this step.

```
# Find redundant rules
subset.matrix <- is.subset(rules.sorted, rules.sorted)
subset.matrix[lower.tri(subset.matrix, diag=T)] <- NA
redundant <- colSums(subset.matrix, na.rm=T) >= 1
which(redundant)
# Remove redundant rules
rules.pruned <- rules.sorted[!redundant]
inspect(rules.pruned)
```

I tried using this because in the image above (the rules I created) I believe there is a lot of redundancy. For instance: 2& 3 are redundant, 3&4, 5 &6, & 7&9 are all redundant to one another. When I tried using this code, All the code ran except for the one I highlighted red and bolded.
I was given this error:
Warning message:
In `[<-`(`*tmp*`, as.vector(i), value = NA) :
x[.] <- val: x is "ngTMatrix", val not in {TRUE, FALSE} is coerced; NA |--> TRUE.

4. Make several graphic displays of your rules. Interpret your results. Include one graph of following:
plot(rules.sorted, method = "graph", engine = "htmlwidget") *Hover over circles to get info about rules.*