Reglas de asociación con Titanic

Contents

Preprocesamiento de datos con el dataset titanic.

> El hundimiento del Titanic es una de las tragedias marítimas más conocidas de la historia. El 15 de abril de 1912, durante su viaje inaugural, el Titanic se hundió después de chocar contra un iceberg. En el accidente murieron 1502 personas de las 2224 que habían embarcado, inluyendo pasajeros y tripulación. Una de las razones por las que no se encontraron más supervivientes fue la falta de espacio en los barcos salvavidas. Así, aunque la suerte sin duda sonrió a los supervivientes, también resultaron más favorecidos algunos grupos de personas, como las mujeres, los niños y los pasajeros de la clase superior.

En este problema analizaremos qué tipos de personas tuvieron más probabilidades de sobrevivir. Para ello, aplicaremos extracción de reglas de asociación que nos permitirán conocer las características de los pasajeros sobrevivieron al hundimiento.

Lectura y preprocesamiento de los datos

Carga de datos

Comenzaremos utilizando el fichero *titanic* de Kaggle, donde encontramos los datos de 891 pasajeros y que utilizaremos para extraer reglas.

Para lectura de datos, utilizaremos alguna de las variantes de la función read. A continuación, podemos inspeccionar el contenido de la tabla de datos, que se almacena en formato tibble.

```
datos <- read_csv('titanic.csv')
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   PassengerId = col_double(),
##   Survived = col_double(),
##   Pclass = col_double(),
##   Name = col_character(),
##   Sex = col_character(),
##   Age = col_double(),
##   SibSp = col_double(),
##   Parch = col_double(),
##   Ticket = col_character(),
```

```
##   Fare = col_double(),
##   Cabin = col_character(),
##   Embarked = col_character()
## )
```

```
head(datos)
```

```
## # A tibble: 6 x 12
##   PassengerId Survived Pclass Name    Sex     Age SibSp Parch Ticket  Fare Cabin
##         <dbl>    <dbl>  <dbl> <chr>   <chr> <dbl> <dbl> <dbl> <chr>  <dbl> <chr>
## 1           1        0      3 Braund~ male     22     1     0 A/5 2~  7.25 <NA>
## 2           2        1      1 Cuming~ fema~    38     1     0 PC 17~ 71.3  C85
## 3           3        1      3 Heikki~ fema~    26     0     0 STON/~  7.92 <NA>
## 4           4        1      1 Futrel~ fema~    35     1     0 113803 53.1  C123
## 5           5        0      3 Allen,~ male     35     0     0 373450  8.05 <NA>
## 6           6        0      3 Moran,~ male     NA     0     0 330877  8.46 <NA>
## # ... with 1 more variable: Embarked <chr>
```

Estado de los datos

Podemos identificar los valores perdidos de la tabla utilizando `df_status()`, del paquete funModeling.

```
df_status(datos)
```

```
##       variable q_zeros p_zeros q_na  p_na q_inf p_inf      type unique
## 1  PassengerId       0    0.00    0  0.00     0     0   numeric    891
## 2     Survived     549   61.62    0  0.00     0     0   numeric      2
## 3       Pclass       0    0.00    0  0.00     0     0   numeric      3
## 4         Name       0    0.00    0  0.00     0     0 character    891
## 5          Sex       0    0.00    0  0.00     0     0 character      2
## 6          Age       0    0.00  177 19.87     0     0   numeric     88
## 7        SibSp     608   68.24    0  0.00     0     0   numeric      7
## 8        Parch     678   76.09    0  0.00     0     0   numeric      7
## 9       Ticket       0    0.00    0  0.00     0     0 character    681
## 10        Fare      15    1.68    0  0.00     0     0   numeric    248
## 11       Cabin       0    0.00  687 77.10     0     0 character    147
## 12    Embarked       0    0.00    2  0.22     0     0 character      3
```

Algunas observaciones interesantes:

- Los valores de *PassengerId* y *Name* son únicos
- Existen dos valores diferentes para *Survived*
- No sobrevivieron 549 pasajeros (61.62%)
- Aparecen numerosos valores perdidos (*na*) en las variables *Age* y *Cabin*
- La mayor parte de los atributos son numéricos

Selección de variables y valores perdidos

Para realizar nuestro análisis no necesitamos todas las variables; nos interesan solamente varias de ellas que *sospechamos* que pueden ser de utilidad:

- `Pclass`: clase del barco en que viajaba el pasajero (1, 2, 3)
- `Age`: edad del pasajero (0, 100)
- `Sex`: sexo del pasajero (male, female)
- `Survived`: sobrevivió o no (1, 0)

Por lo tanto, podemos seleccionar solo las que nos interesan:

```
datos_seleccion <- datos %>%
  select(Pclass, Age, Sex, Survived)
head(datos_seleccion)
```

```
## # A tibble: 6 x 4
##   Pclass   Age Sex    Survived
##    <dbl> <dbl> <chr>     <dbl>
## 1      3    22 male          0
## 2      1    38 female        1
## 3      3    26 female        1
## 4      1    35 female        1
## 5      3    35 male          0
## 6      3    NA male          0
```

Podemos obviar las filas que contienen valores perdidos con `drop_na()`. También podrían aplicarse otros procedimientos más sofisticados para tratar con esos valores perdidos (ver paquete `mice`.

```
datos_seleccion <- datos_seleccion %>%
  drop_na()
head(datos_seleccion)
```

```
## # A tibble: 6 x 4
##   Pclass   Age Sex    Survived
##    <dbl> <dbl> <chr>     <dbl>
## 1      3    22 male          0
## 2      1    38 female        1
## 3      3    26 female        1
## 4      1    35 female        1
## 5      3    35 male          0
## 6      1    54 male          0
```

Recodificación y discretización de variables

Para trabajar con reglas de asociación, necesitamos que las variables del problema sean de tipo cuantitativo. Por lo tanto, necesitamos ajustar los valores de `Pclass`, `Age` y `Survived`.

Para `Pclass`, únicamente vamos a recodificar los valores usando `mutate()` y la instrucción condicional `ifelse()`:

```
# Survived
datos_cuantitativos <- datos_seleccion %>%
  mutate(Survived = ifelse(Survived == 0, "no", "yes"))
```

Para `Age` y `Survived`, el procedimiento es similar, usando condicionales más complejos como `case_when()`:

```
# Pclass
datos_cuantitativos <- datos_cuantitativos %>%
  mutate(Pclass = case_when(
    Pclass == 1 ~ '1st',
    Pclass == 2 ~ '2nd',
    Pclass == 3 ~ '3rd')
  )

# Age
datos_cuantitativos <- datos_cuantitativos %>%
  mutate(Age = case_when(
    Age >= 18 ~ 'adult',
```

```
    TRUE ~ 'child')
  )

head(datos_cuantitativos)
```

```
## # A tibble: 6 x 4
##   Pclass Age   Sex    Survived
##   <chr>  <chr> <chr>  <chr>
## 1 3rd    adult male   no
## 2 1st    adult female yes
## 3 3rd    adult female yes
## 4 1st    adult female yes
## 5 3rd    adult male   no
## 6 1st    adult male   no
```

Transformación a formato transaccional

Para ejecutar el algoritmos Apriori necesitamos convertir nuestra tabla a formato transaccional. En este problema, los ítems serán las características de los individuos expresadas en formato `<atributo=valor>`. Por ejemplo, la transacción correspondiente al primer individuo será:

`<Pclass=3rd>, <Age=adult>, <Sex=male>, <Survived=no>`

Para ello, necesitamos primero pasar todas las columnas a tipo de dato `factor`:

```
datos_cuantitativos <- datos_cuantitativos %>%
  mutate_all(as.factor)
head(datos_cuantitativos)
```

```
## # A tibble: 6 x 4
##   Pclass Age   Sex    Survived
##   <fct>  <fct> <fct>  <fct>
## 1 3rd    adult male   no
## 2 1st    adult female yes
## 3 3rd    adult female yes
## 4 1st    adult female yes
## 5 3rd    adult male   no
## 6 1st    adult male   no
```

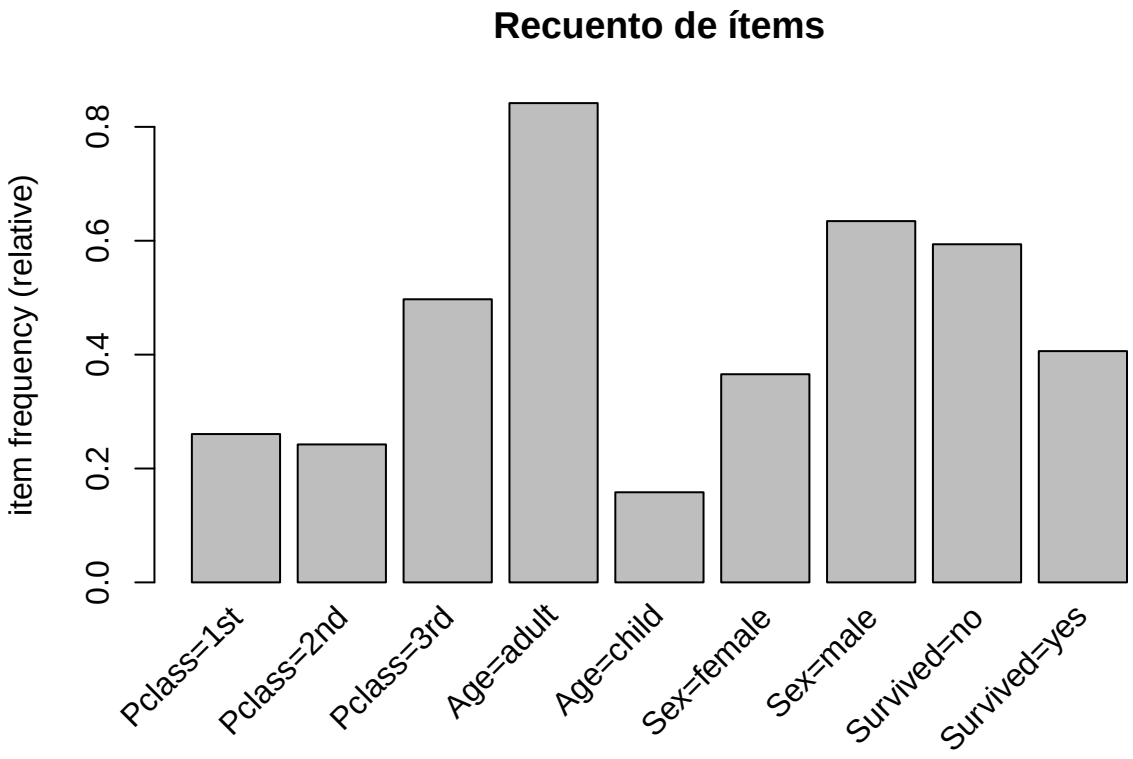
Y, a continuación, convertir a `"transactions"` con la función `as()`:

```
datost <- as(datos_cuantitativos, "transactions")
datost
```

```
## transactions in sparse format with
##  714 transactions (rows) and
##  9 items (columns)
```

```
# inspect(datost)

itemFrequencyPlot(datost, type="relative", main="Recuento de ítems")
```



Ejercicios

1. Obtén los itemsets frecuentes de los datos.
2. Ejecuta el proceso de extracción de reglas usando:

- Soporte: 1%, Confianza: 70%
- Soporte: 1%, Confianza: 85%

3. Visualiza una selección de las 20 reglas con mayor soporte.
4. Comenta varias reglas obtenidas que te parezcan interesantes en términos de *lift*.
5. Extrae reglas que se refieran solamente a pasajeros adultos. ¿Qué pasaría si, en lugar de aplicar las instrucciones de cesta.Rmd, hiciéramos un `filter(Age <= 10)` sobre `datos_cuantitativos` para quitar los pasajeros menores de edad?