Práctica 3: Listas y data frames

A continuación proponemos distintas tareas relacionadas con la creación y manipulación de listas y data frames. Crea un fichero script con el código que permita resolverlas, incluyendo en el mismo los comentarios que estimes oportunos. Este script deberás enviarlo a través de PRADO siguiendo las instrucciones proporcionadas en la tarea allí creada.

1. Crea un objeto de tipo lista con estas tres componentes: $x1 = (1, 2, 3, 4, 5)$, $x2 = (2, 3, 4, 5, 6)$ y $x3 = (3, 4, 5, 6, 7)$. A partir de ella resuelve las siguientes tareas:

   a) Crea un vector `x` con una muestra de 10 números aleatorios de una distribución uniforme en el intervalo (0,1). Añade dicho vector como una nueva componente a la lista anterior.

   b) Crea un vector `y` con una muestra de 10 números aleatorios de una distribución normal estándar. Añade dicho vector como una nueva componente a la lista anterior.

   c) Utiliza la función `lapply` para calcular la suma de cada componente de la lista. Observa qué tipo de objeto devuelve. Después prueba con la variante `sapply`, ¿qué diferencia observas entre las dos funciones?.

   d) Escribe el siguiente código:

   ```
   reg<-lm(y~x)
   ```

   [1] y utiliza una función adecuada para confirmar que `reg` es un objeto de tipo lista.

   e) Utiliza una función adecuada para obtener qué tipo de objetos constituyen las componentes de `reg`.

   f) Crea una matriz que contenga por columnas las componentes `residuals` y `fitted.values` del objeto `reg`, además de los vectores `x` e `y`. Añade nombres a las columnas de dicha matriz.

2. A veces los datos que tenemos para un análisis estadístico corresponden a datos agregados en forma de tabla de frecuencias. Crea un data frame con nombre `datos` con los datos que aparecen a continuación:

x_i	y_i	n_i
1.2	15	12
1.8	18	23
2.2	10	5
2.5	12	9
1.1	16	11

[1] La función `lm` permite estimar un modelo de regresión lineal. En este caso sería de regresión lineal simple ($y_i = \beta_0 + \beta_1 x_i + \epsilon_i$, $i = 1, \ldots, n$) con los elementos de `x` como observaciones de la variable explicativa, y los de `y` como observaciones de la variables de respuesta.

Se trata de datos agregados donde las dos primeras columnas corresponden a los valores que se observan en una muestra de dos variables estadísticas, (x_i, y_i), y la última columna contiene la frecuencia absoluta (n_i), esto es, el número de veces que se observa el par (x_i, y_i). De este modo el tamaño de la muestra (n) es la suma de dichas frecuencias absolutas. A partir de dicho data frame realiza las siguientes tareas:

a) Calcula el tamaño de la muestra.

b) Calcula las media aritméticas de las observaciones de las variables $\bar{x}$ e $\bar{y}$, así como las cuasivarianzas, s_x^2 y s_y^2.

c) Crea un segundo data frame con nombre `datos.n` que recoja las n observaciones individuales por filas, esto es, repitiendo las filas de `datos` tantas veces como indique la columna de la frecuencia absoluta.

d) A partir del data frame `datos.n` calcula de nuevo las medias aritméticas y las cuasivarianzas (usando `mean` y `var`, respectivamente) y comprueba el resultado anterior con los datos agregados.

e) La tipificación de los datos es una práctica habitual y requerida en algunas técnicas estadísticas. Consiste en una transformación del tipo $z_i = (x_i - \bar{x})/s_x$, de modo que la media de los z_i es 0 y su cuasi-varianza es 1. Añadir dos columnas al final del data frame `datos.n` con los valores tipificados de las variables x e y. Realiza esta tarea de dos formas, primero utilizando la función `transform` y luego utilizando `within`.

3. En este ejercicio vamos a realizar varias manipulaciones sobre el data frame `ChickWeight` del paquete *datasets*. Comienza escribiendo `help(ChickWeight)` y descubre el tipo de datos que contiene el data frame. Después resuelve las siguientes tareas:

a) Imprime en la venta de la consola las primeras 5 filas del data frame `ChickWeight` y las 3 últimas, utilizando para ello las funciones `head` y `tail`, respectivamente.

b) Imprime la estructura del objeto `ChickWeight`.

c) Realiza un resumen descriptivo numérico elemental de todas las variables del data frame con `summary`.

d) Realiza el mismo tipo de resumen pero ahora solo de la variable `weight` para los distintos niveles del factor dieta, usando la función `tapply`. Almacena el resultado en un objeto con nombre `peso.dieta`. ¿Qué tipo de objeto es `peso.dieta`?[2]

e) Crea un data frame (`peso.dieta.2`) colocando por columnas el resumen obtenido del peso para cada tipo de dieta. Cada columna tendrá como nombre el de la correspondiente medida descriptiva ("Min.", "1st Qu.", etc.).

[2]Si inspeccionas la ayuda de la función `tapply` entenderás mejor el resultado.

f) La función `aggregate` permite resumir columnas de un data frame para cada uno de los niveles de un factor[3]. Utiliza esta función para realizar el mismo resumen que realizaste antes en el objeto `peso.dieta`. ¿Qué tipo de objeto devuelve esta función? Vuelve a crear el data frame `peso.dieta.2` con la estructura especificada antes a partir del objeto que devuelve `aggregate`.

g) Crea un data frame (`Chick100`) con una submuestra de los datos contenidos en `ChickWeight` seleccionando aleatoriamente (sin reemplazo) 100 filas[4].

h) Muestra el data frame `Chick100` con sus columnas permutadas aleatoriamente[5].

i) Muestra el data frame `Chick100` con sus columnas por orden alfabético.

j) Muestra los datos del data frame `Chick100` ordenados según la variable `Diet` (orden ascendente). Observa que cómo trata R los empates en dicha ordenación. Repite la operación rompiendo los empates de acuerdo al valor en la variable `Weight`.

k) Extrae del data frame `Chick100` una submuestra conteniendo solo una observación para cada tipo de dieta (variable `Diet`), en concreto la que corresponda al mayor valor de la variable `weight`. [Sugerencia: ordena las filas del data frame según `weight` en orden descendente, después puedes usar la función `duplicated`[6] aplicada a columna `Diet` para quedarse solo con la primera observación correspondiente a cada tipo de dieta.]

[3]Por ejemplo supongamos un data frame `df` con varias columnas donde la segunda de ellas es un factor, si queremos calcular las medias de la primera columna para los distintos niveles del factor entonces podríamos escribir `aggregate(df[,1],by=list(df[,2]),mean)`

[4]Seleccionar aleatoriamente observaciones de una muestra forma parte de técnicas estadísticas y del Machine Learning como el bootstrap y validación cruzada (*cross-validation*).

[5]Esta operación es necesaria en técnicas estadísticas relacionadas con big data y Machine Learning como *random forest*.

[6]Esta función aplicada a un vector devuelve (`duplicated(v)`) un vector lógico (del mismo tamaño del original, `v`) indicando con `TRUE` las posiciones del vector que contienen el mismo valor. De este modo `!duplicated(v)` se puede usar como filtro para seleccionar las filas del data frame en este ejercicio eliminando las duplicaciones.