Creación de paquetes en R

by Román Salmerón Gómez (18/04/2023)

Paso 1: Crear el paquete

1.- Crear un archivo con extensión *.R* donde aparezcan las funciones que van a formar el paquete y se carguen los datos (si los hubiese) que van a estar disponibles en el mismo (ya sea para usarlos en los ejemplos como por cualquier otro motivo). Lo ideal es llamarlo con el nombre que vaya a tener el paquete, por ejemplo, *mipaquete.R*.

2.- Abrir dicho archivo con RStudio y ejecutar todo su contenido. A continuación ejecutar *package.skeleton("mipaquete")*:

a. Se creará una carpeta llamada 'mipaquete', donde a su vez hay las siguientes tres carpetas: 'data' (contiene bases de datos si es que las hemos incluido), 'man' (contiene los archivos *.Rd* de ayuda, que hay que editar) y 'R' (contiene las funciones del paquete).
b. También se crean tres archivos: 'DESCRIPTION' (aquí describiremos las principales características del paquete y especificaremos los paquetes que son necesarios mediante la opción *Depends*), 'NAMESPACE' (aquí especificaremos también la carga de paquetes mediante *import*, la carga de funciones que son necesarios mediante *importFrom* (¡Ojo! Cuando se chequea el paquete se sugiere qué funciones incluir con *importFrom*) y las funciones a exportar mediante *export*) y 'Read-and-delete-me' (leer y eliminar).

3.- El siguiente paso es editar el contenido de los archivos *.Rd* de la carpeta 'man'. Estos archivos son la ayuda del paquete. Si se editan con RStudio, cosa que recomiendo, hay una pestaña en la parte de arriba llamada 'Preview' que permite previsualizar el resultado y comprobar que no hay errores.

4.- Igualmente hay que editar los archivos 'DESCRIPTION' y 'NAMESPACE'. En este caso, es suficiente con usar el Bloc de Notas. Si en el segundo no escribimos nada, los pasos dados al depurar el paquete nos ofrecerán información de qué añadir.

Nota: para saber qué escribir en los dos últimos puntos puede ser interesante echar un vistazo al contenido de las dos siguientes webs: https://es.scribd.com/document/327287534/Crear-paquetes-R-pdf# y https://oscarperpinan.github.io/R/Paquetes.pdf.

Paso 2: Depurar el paquete

5.- Una vez finalizado el proceso de creación del paquete, hay que chequearlo usando el símbolo del sistema mediante el comando *R CMD check --as-cran mipaquete*. Evidentemente previamente hay que ir a la carpeta donde está el paquete (la carpeta creada al ejecutar el comando *package.skeleton* vamos). Lo mejor es crear una carpeta en C:\ llamada (por ejemplo) *PaquetesR* y copiar la carpeta creada con *package.skeleton* en ella. A continuación, con el comando *cd..* se llega hasta ella y entonces ejecutar el código anterior. Se crea entonces la carpeta *mipaquete.Rcheck*:

- Si hay errores, en esta carpeta se crea un archivo *.log* donde podemos ver exactamente lo que sale en el símbolo del sistema. También se tiene un archivo *.out* donde salen qué errores hay para poder corregirlos.
- Si no, genera el paquete.

6.- Cuando esté depurado el paquete, lo construyo mediante el comando *R CMD build mipaquete*, de forma que se crea *mipaquete.tar.gz*. A continuación chequearlo con el comando *R CMD check –as-cran mipaquete.tar.gz*. Es posible que se corrijan *NOTEs* que han aparecido en el chequeo previo (se machaca la carpeta *mipaquete.Rcheck*).

Nota: Para que el proceso de depuración funcione hay que tener R en el PATH de Windows, en caso contrario todo lo descrito relacionado con el símbolo del sistema no funcionará. Así, busco R en mi ordenador, *C:\Program Files\R\R-4.2.2\bin*, y lo añado al PATH de la siguiente forma: en *Panel de Control* buscamos la opción *Configuración avanzada del sistema* y en la pestaña que sale nos centramos en *Variables de entorno*. En *Variables del sistema* buscamos PATH y lo editamos añadiendo la ruta anterior.

Paso 3: Publicar el paquete

7.- Opción 1 (la más fácil): Subo el archivo *mipaquete.tar.gz* directamente a CRAN en el enlace https://cran.r-project.org/submit.html. Puesto que hemos depurado el paquete, no deberíamos tener problemas en que lo publicasen, aunque siempre es posible que nos requieran cambiar algo.

8.- Opción 2 (la más elaborada): Subirlo a RFORGE (https://r-forge.r-project.org/) y, una vez que esté aquí disponible, trasladarlo a CRAN: manual de instrucciones disponible en http://download.r-forge.r-project.org/R-Forge_Manual.pdf (no es algo inmediato). Al igual que antes, puesto que hemos depurado previamente el paquete, no deberíamos tener excesivos problemas.

9.- Opción 3: Subirlo a GitHub (https://github.com/). Una posibilidad similar a esta es compartir directamente el archivo *mipaquete.tar.gz* generado en el paso anterior para instalarlo directamente en R/RStudio. ¿Cómo se haría?

- Darse de alta en GitHub: https://github.com/login.
- En el icono de nuestro perfil en la parte superior derecha, seleccionar la opción de *Your repositories*. A continuación, pinchar en *New* para crear un nuevo repositorio y rellenar la información que nos piden: nombre del repositorio (en mi caso, mipaquete), breve descripción (opcional), si queremos que sea público o privado (si deseamos que lo use cualquiera, debemos elegir público) o si queremos que se incialice con un archivo README (aquí https://github.com/rasmusab/bayesian_first_aid un ejemplo de lo bien que puede quedar). Finalmente pinchar sobre *Create repositorie*. Se crea el repositorio público https://github.com/rnoremlas/mipaquete.
- En el repositorio creado, la parte central saldrá el mensaje *Get startet by creating a new file or uploading an existing file*. Elegimos la segunda opción de forma que podemos subir archivos/carpetas sin más que arrastrarlas a la zona donde pone *Drag additional files here to add them to your repository*. En nuestro caso, subiremos el contenido de la carpeta *mipaquete* existente dentro de la carpeta *00_pkg_src* de la carpeta *mipaquete.Rcheck* creada en el paso sexto anterior (también se podría elegir el

contenido de la carpeta que hemos depurado en el paso quinto). Finalmente pinchamos en *Commit changes*.

- Para instalarlo necesitamos instalar y cargar el paquete *remotes* y luego ejecutar el código *install_github(rnoremlas/mipaquete)*. Adviértase que para poder instalar el paquete creado se necesita tener instalado Rtools 4.2 (se puede descargar de la web https://cran.r-project.org/bin/windows/Rtools/rtools42/rtools.html).

Nota: De las opciones propuestas, la más inmediata es la primera. Al mismo tiempo, no aconsejo instalar paquetes usando la tercera opción a no ser que se tenga total confianza en el desarrollador del paquete.