Desarrollo de un entorno gráfico para R y ULLRtoolbox.

Development of a graphical user interface to R and ULLRtoolbox

Domingo Yeray Rodríguez Martín
Departamento de Ingeniería Informática
Escuela Técnica Superior de Ingeniería Informática

Trabajo de Fin de Grado

La Laguna, 10 de junio de 2014

D. **Jesus Miguel Torres Jorge**, con N.I.F. 43826207-Y profesor Contratado Doctor de Universidad adscrito al Departamento de Ingeniería de Sistemas y Automática y Arquitectura y Tecnología de Computadores de la Universidad de La Laguna

C E R T I F I C A

Que la presente memoria titulada:

"Desarrollo de un entorno gráfico para R y ULLRtoolbox."

ha sido realizada bajo su dirección por D. Domingo Yeray Rodríguez Martín, con N.I.F. 78628933-Y.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 10 de junio de 2014

Agradecimientos

En primer lugar me gustaría agradecer al tutor del proyecto, Jesus Miguel Torres, por facilitarme las herramientas necesarias para comenzar y desarrollar dicho proyecto, así como por su predisposición a ayudarme a resolver las dudas en cualquier momento.

También agradecer a Juan Andrés Hernández por su interés mostrado en este proyecto y facilitarme la documentación necesaria sobre su script ULLRtoolbox.

Por último, a Aurora Padilla, antigua alumna de psicología, por explicarme los entresijos del análisis estadístico.

Resumen

Los programas informáticos se han convertido, hoy en día, en una herramienta básica utilizada por el análisis estadístico como apoyo fundamental a la hora de realizar diferentes operaciones y para facilitar una mayor comodidad a los usuarios. Actualmente, R es uno de los programas más utilizados debido a su potencia y a su distribución como software libre. Sin embargo, cuenta con dos grandes inconvenientes: tiene una sintaxis compleja y no dispone de interfaz gráfica que permita ser usable por aquellos que no son informáticos. Para solucionar el primer inconveniente, el Departamento de Psicobiologia y Metodología de la Universidad de La Laguna ha creado ULLRToolbox, un script en el que se pueden llamar a distintas funciones de análisis estadístico de forma simple. Este proyecto pretende solventar el segundo inconveniente creando una interfaz gráfica que permita llamar mediante menús a las distintas funciones del script ULLRToolbox haciendo uso de la biblioteca Qt y en concreto un binding de éste llamado PySide.

Palabras clave

R, ULLRToolbox, Qt, PySide, RPy2, interfaz gráfica, Python.

Abstract

Computer programs have become, nowadays, in a basic tool using by statistical analysis as essential support at the time of doing different operations and to facilitate a convenience for users. Currently, R is one of the most used program due to its power and to its distribution as free software. However, it counts on two disadvantages: complex syntax and lack of a GUI that allows to be usable by people are not computer programmers. To solve the first diadvantage, the Psychobiology and Methodology Department of the University of La Laguna has created ULLRToolbox, a script in which different functions of stadistic analysis can be called in an easy way. This Project pretends to solve the second disadvantage, developing a GUI that allows to call differents functions of the ULLRToolbox script, trough menus, using Qt libraries, and in particular, a binding called PySide.

Keywords

R, ULLRToolbox, Qt, PySide, RPy2, graphical user interface (GUI), Python.

Índice general

Índice de Figuras

Índice de Tablas

CAPÍTULO 1

Introducción

Desde hace unos años, el análisis estadístico ha visto como surgían decenas de herramientas que permiten a profesionales e investigadores de este campo, solucionar diversos problemas de forma más rápida y eficiente que realizando estos cálculos de forma manual.

Estos programas no solucionan automáticamente una planificación deficiente del problema, ya que es el propio investigador el que debe elegir una técnica de análisis adecuada en función de los objetivos de la investigación. De ejecutar estas técnicas de análisis es de lo que se ocupa el software estadístico.

El programa estadístico más conocido a nivel global es el denominado inicialmente *Statistical Package for the Social Sciences*, más conocido por sus siglas SPSS. Sin embargo, en el año 2009 IBM adquirió este software por lo que el programa ha pasado a denominarse IBM SPSS aunque comúnmente se le suele llamar por sus siglas iniciales. Es el programa de análisis estadístico más conocido y usado por una cuestión principal, posee una *sencilla interfaz* para la mayoría de los análisis por lo que permite que personas puedan aprender a usarlo sin las grandes dificultades que tienen otros programas con los que compite.

El mayor inconveniente de SPSS es que su uso requiere la compra de una licencia que provoca que en muchas instituciones no puedan permitir su coste elevado. Asimismo, actualmente compite con software bajo licencia como lo son SAS, MATLAB, Statistica o Stata entre otros. Sin embargo, su mayor competidor es el Lenguaje R.

R es un lenguaje y entorno de programación para análisis estadístico y gráfico. Se trata de un proyecto de software libre y se distribuye bajo licencia GNU GPL. Este lenguaje se encuentra disponible para los sistemas operativos Windows, Mac, Unix y Linux. Al tratarse de un lenguaje de programación, permite que los usuarios puedan extenderlo añadiendo sus propias funciones. De hecho, gran parte de las funciones de R están escritas en el mismo R.
La gran *ventaja* de R sobre SPSS es que el primero es libre y por lo tanto, gratuito, por lo que su uso está bastante extendido. Sin embargo, tiene dos grandes *inconvenientes*

respecto a SPSS. El primer inconveniente es que requiere de un cierto esfuerzo para aprender las reglas fundamentales de su sintaxis orientada a objetos y el segundo inconveniente es que no posee interfaz gráfica por lo que es menos usable y cómodo para el usuario.

Para solventar el problema de la elevada curva de aprendizaje del lenguaje, desde la Universidad La Laguna y, en concreto, el departamento de Psicobiologia y Metodología ha creado *ULLRToolbox* que es una caja de herramientas que viene a aliviar al usuario precisamente de la necesidad de aprender dicho lenguaje y su más o menos compleja sintaxis. A partir de llamadas a funciones muy simples incluso el usuario novel puede llevar a cabo análisis de datos desde los procedimientos más simples de estadística descriptiva hasta los más complicados de análisis multivariado.

ULLRToolbox hace un uso intensivo de las librerías stats, MASS, nlme, lme4, car, psych, languageR, bestglm, foreign, klar, gplots, GPArotation, ca, mice, polycor, corpcor, ltm, vcdExtra, relaimpo y candisc. La mayor parte de las funciones integran de forma automatiza las múltiples librerías mencionadas anteriormente con el objetivo de que en un solo análisis permita ejecutar una salida de resultados integrada más acorde a la obtenida en los paquetes comerciales al uso. El toolbox pretende lograr que el usuario no familiarizado con los lenguajes de programación pueda, de forma gratuita, rápida y simple, solicitar un análisis gráfico estadístico completo con la llamada a una única función que le devuelve los resultados agrupados en una sola salida.

El otro problema de R y que se ha mencionado anteriormente es que, debido a que es un lenguaje de programación, no dispone de interfaz gráfica. Sin embargo, existen diversas interfaces gráficas creadas por la comunidad de desarrolladores como JGR, R commander, RExcel, rggobi, RKWard, Sage, Rstudio,… entre muchos otros. Asimismo, la funcionalidad de R puede ser invocada desde código desarrollado en otros lenguajes de script como Python (mediante RPy) y Perl (mediante Statistics::R).

Por lo tanto, los dos inconvenientes principales de R tienen una solución independiente. Por un lado tenemos que el problema del aprendizaje del lenguaje ha sido solventado con ULLRToolbox y la ausencia de una interfaz intuitiva con programas externos como los mencionados anteriormente. El *Trabajo de Fin de Grado* que hace alusión la presente memoria intenta unir ambas soluciones para que ULLRToolbox se convierta en una verdadera alternativa a SPSS.

El *Trabajo de Fin de Grado* consiste en crear desde cero una aplicación de escritorio que disponga de una interfaz gráfica que permita llamar mediante menús y cuadros de diálogos las principales funciones del script ULLRToolbox.

CAPÍTULO 2

Estado de la técnica

Hoy día existen múltiples herramientas para crear aplicaciones con interfaz gráfica. Prácticamente por cada lenguaje de programación existe una biblioteca o librería que permite desarrollar cómodamente en el lenguaje que más guste al usuario.

El sistema operativo más usado en el mundo es Windows y por ende, el lenguaje más conocido para desarrollar interfaces gráficas es C# y el framework .NET. Sin embargo, tiene el inconveniente de que es software comercial distribuido por Microsoft y no es multiplaforma como otras alternativas. Para solucionar estos dos inconvenientes, ha surgido un proyecto de software libre llamado Mono que crea un grupo de herramientas libres, basadas en GNU/Linux y compatibles con .NET.

El lenguaje C# fue creado y desarrollado por Microsoft para hacer frente a Java. La gran ventaja de este último es que lo que escribes una vez puedes ejecutarlo en cualquier plataforma. Java provee dos API's con las que podemos trabajar para desarrollar GUIs, la más básica es AWT (Abstrct Window Toolkit). Las más desarrolladas se hacen con Swing. Sin embargo, no ofrece un gran rendimiento en aplicaciones de escritorio y presentan más problemas debido a que están hechos en Java puro. Además, su interfaz no siempre se corresponde a la propia del sistema operativo donde fue diseñada.

En los entornos de software libre disponemos de tres bibliotecas multiplataformas que se encuentran a la cabeza en el desarrollo de aplicaciones de escritorio. Estas bibliotecas son wxWidgets, GTK+ y Qt.

Las bibliotecas *wxWidgets* permite desarrollar interfaces gráficas en C++. Están publicadas bajo una licencia LGPL, similar a GPL con la excepción de que el código del ejecutable enlazado con ellas, puede ser propietario, permitiendo desarrollar aplicaciones empresariales sin coste de licencias. Proporcionan una interfaz gráfica basada en las bibliotecas ya existentes en el sistema (nativas), con lo que se integran de forma óptima y resultan muy portables entre distintos sistemas operativos. También pueden ser utilizadas en otros lenguajes de programación como Java, JavaScript, Perl, Python, Smalltalk o Ruby.

GTK+ o The GIMP Toolkit es un conjunto de bibliotecas multiplaforma para desarrollar interfaces gráficas de usuario (GUI), principalmente para los entornos gráficos GNOME, XFCE y ROX aunque también se puede usar en el escritorio de Windows, Mac OS y otros. Inicialmente fueron creadas para desarrollar el programa de edición de imagen GIMP, sin embargo, actualmente se usan bastante por muchos otros programas en los sistemas GNU/Linux. Junto a Qt es una de las bibliotecas más populares para X Window System. GTK+ se ha diseñado para permitir programar con lenguajes como C, C++, C#, Fortran, Java, Ruby, Perl, PHP o Python.

Por último, tenemos a Qt que es desarrollada a través de Qt Project, donde participa tanto la comunidad, como desarrolladores de Digia y otras empresas. Anteriormente, era desarrollado por la división de software de Qt de Nokia, que entró en vigor después de la adquisición por parte de Nokia de la empresa noruega Trolltech, el productor original de Qt. Años más tarde, Digia adquirió las licencias comerciales de Qt exclusivamente.

Utiliza C++ de forma nativa pero puede ser utilizado por otros lenguajes de programación mediante bindings *(Tabla 1).* Un binding es una adaptación de una biblioteca para ser usada en un lenguaje de programación distinto de aquel en el que ha sido escrita.

Binding	**Descripción**
PyQt	Bindings GPL/Comercial para Python.
PySide	LGPL bindings para Python de OpenBossa.
PythonQt	LGPL bindings para Python
Qyoto	Bindings para C# u otros lenguajes.NET
QtRuby	Bindings para Ruby
Qt Jambi	Bindings para Java
QtAda	Bindings para Ada
FreePascal Qt4	Bindings para Pascal
Perl Qt4	Bindings para Perl
PHP Qt	Bindings para PHP
Qt Haskell	Bindings para Haskell
lqt	Bindings para Lua
DaoQt	Bindings para Dao
QtD	Binding para D

Tabla 1. Listado de los bindings que pueden ser usados en Qt.

Los bindings más destacados son los relacionados con el lenguaje Python. Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que

favorezca un código legible. Es un lenguaje interpretado, usado tipado dinámico y es multiplataforma. Es administrado por la Python Software Foundation. Posee una licencia de código abierto denominada Python Software Foundation License.

Las ventajas de Python es que es rápido de desarrollar, es sencillo, es veloz, soporta varias bases de datos y sus bibliotecas hacen gran parte del trabajo. El inconveniente es que los lenguajes interpretados son más lentos que los compilados, sin embargo, el código de interacción con el usuario no requiere de mucha CPU por lo que el rendimiento no debe ser una preocupación.

PyQt y Pyside son los dos bindings de Qt para desarrollar en Python. PyQt es desarrollado bajo la empresa británica Riverbank Computing mientras que PySide fue liberado en agosto de 2009 por Nokia. Ambas opciones incluyen la licencia LGPL. Estos dos bindings soportan los principales sistemas operativos como Linux, Mac OS X y Windows.

CAPÍTULO 3

Requisitos del proyecto

El requisito general del proyecto consiste en el desarrollo de una interfaz gráfica que permita llamar mediante menús y cuadros de diálogos a las distintas funciones del script ULLRToolbox.

Para establecer los requisitos más concretos de la aplicación, se ha contactado con el desarrollador del script para que nos dé su visión de cómo podría ser la interfaz. Gracias a ellos hemos podido establecer que el programa dispondría de cuatro grandes apartados denominados 'Archivo', 'Datos', 'Análisis' y 'Gráficas', además de añadir otro apartado más llamado 'Acerca de' en el que se detalla la información relevante del programa.

El primer apartado de '*Archivo'* permite la lectura de archivos externos en formato de R, Excel, SPSS y texto plano así como la exportación de los data sets (conjunto de datos mapeados) en formato de R o también en texto plano. La pestaña de *'Datos'* recoge todas aquellas funciones que permiten manipular las bases de datos o las variables. Este apartado es el más extenso ya que tiene funciones como la de agregado, segmentado y todas las que permiten manipular variables como crear una nueva variable, recodificarla, transformarla, tipificar, discretizar y extraer muestras. En el apartado *'Análisis'* irán las técnicas de estadística descriptiva, correlación, contrastes de medias, análisis de varianza (ANOVA) intergrupo, intragrupo o Split-plot, correlación y regresión múltiple entre otras. La última opción llamada *'Gráficas'* constaría de todas las funciones gráficas de las que dispone actualmente el toolbox como diagrama de cajas, histograma, grafica de dispersión, etc.

Una vez establecido el sistema de menús, se determinó que el programa tenga dos pestañas en la ventana central con el objetivo de proporcionar al usuario un apartado de visualización y otro de interacción.

En la pestaña principal se muestra una cuadrícula al estilo excel o SPSS en la cual se puede visualizar de forma clara el dataset abierto por el usuario situando en las columnas la totalidad de las variables y en cada una de las filas los valores obtenidos por los sujetos en cada una de éstas.

En la otra pestaña tendremos tres partes diferenciadas. Por un lado se mostrará un visor con el listado de las variables del data set abierto que permitirá al usuario tratar con las distintas variables sin necesidad de ejecutar comandos (por ejemplo, cambiar el nombre). Por otro lado tendremos dos secciones interconectadas, una de ellas consiste en una región de edición de código que permite escribir nuestro propio código R o funciones del propio ULLRToolbox y ejecutarlas. Además, los resultados, tanto de las funciones llamadas a través del menú como del código que introduzcamos a mano, se pueden ir visualizando de forma interactiva en una sección contigua a la anterior.

Por último, cabe destacar que esta segunda pestaña guarda especial similitud con la interfaz del MATLAB, que es una herramienta de software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M).

CAPÍTULO 4

Solución adoptada

Una vez analizado el estado actual de las herramientas disponibles para desarrollar una interfaz gráfica y haber establecido los requisitos que debe tener el programa se procedió a adoptar una solución para abordar el problema.

En primer lugar se eligió la biblioteca para realizar la interfaz gráfica. Uno de los requisitos del proyecto era que programa a desarrollar debería poder ser multiplataforma por lo que la opción de usar un entorno de C# y .NET como el comentado en el estado de la técnica quedó descartada desde el inicio. Se optó por software libre por lo que las opciones de Qt, wxWidgets y GTK+ fueron las contempladas, siendo la primera la elegida debido a su gran facilidad de uso, su gran apoyo y la variedad de bindigs que se pueden usar que dan una libertad al desarrollador para elegir el lenguaje con el que se sienta más cómodo.

El lenguaje elegido fue Python por varios motivos:

- Por lo general en Python se necesitan menos líneas para hacer lo mismo que en C++, por lo que es más productivo en poco tiempo.
- Como Java, Python se encarga de la gestión de memoria y no tiene ni referencias ni punteros.
- Y por último y el más relevante, existe un binding de Python para R (RPy) que facilita la ejecución de los comandos y las funciones de ULLRToolbox.

Una vez elegido el lenguaje, había que decidir por uno de los bindings de Qt para Python. Los dos que se probaron fueron PyQt y PySide. Se empezó a desarrollar la aplicación en PyQt pero los errores en la instalación del mismo y la escasa documentación a la hora de abordar un problema provocaron que se probase PySide. PySide dispone de una documentación más detallada dentro de la propia página oficial de Qt Project, además de la existencia de múltiples tutoriales externos de alta calidad.

La última versión de PySide es la 1.2.2 lanzada el 25 de Abril de 2014 y provee acceso completo a la versión 4.8 del framework de Qt. Mientras que para interpretar el código se usó Python 2.7.

Tal y como se comentó anteriormente, uno de los motivos por el cual se eligió Python como lenguaje de desarrollo es que dispone de un binding de R llamado RPy. RPy permite disfrutar de la elegancia de la programación en Python proveyendo acceso de forma simple a las capacidades gráficas y estadísticas de R. Actualmente existe una versión llamada RPy2 que consiste en un rediseño de RPy proporcionando nuevas funcionalidades como el acceso a bajo nivel de R.

Qt provee un IDE llamado Qt Creator que ha sido creado por Trolltech y que es multiplataforma soportando los sistemas operativos principales como GNU/Linux, Mac OS X y Windows XP o superiores. Esta herramienta está diseñada para desarrollar un proyecto hecho en Qt con C++ y provee muchas funcionalidades para ejecutar el código y depurarlo aunque también permite crear clases en Python de forma menos integrada en el IDE. Sin embargo, usamos un editor de texto convencional llamado *Sublime Text 2* el cual permite instalarle scripts que permiten que el desarrollo en Python y en PySide sea bastante cómodo.

Para crear las interfaces gráficas si se ha usado una herramienta que provee Qt Creator, llamado *Qt Designer*. Esta herramienta permite crear los widgets, las ventanas y los diálogos de la aplicación siendo posible editarlos directamente desde el editor gráfico. Estos widgets pueden ser accedidos desde el código utilizando las señales de Qt. El formato de estos es un archivo QRC, el nombre completo de este formato de archivo es *Qt Resource Collection File*.

En PySide no podemos acceder a este tipo de archivos directamente como en C++ sino que hay que hacer un paso intermedio y para ello hay dos soluciones. Por un lado podemos compilar estos archivos qrc mediante un comando llamado pyside-rcc. Sin embargo, si tenemos muchos archivos como es el caso de este proyecto resultaría tedioso compilarlo cada vez que modifiquemos la interfaz del programa. Por eso, existe otra solución más cómoda y consiste en utilizar una herramienta de PySide llamada QUiLoader que permite ahorrarnos convertir dichos archivos. Se detallará mejor el funcionamiento de esta herramienta en el capítulo de los detalles de la implementación del proyecto.

Otra herramienta usada en el desarrollo del proyecto fue git. Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Permite mantener una copia de nuestro código en un servidor de forma rápida mediante la línea de comandos. El servidor git usado fue Github y en concreto el repositorio de la Escuela Técnica de Ingeniería Informática de la Universidad de La Laguna. Desde Github cualquier persona puede descargarse el código y ejecutar el software desarrollado, además de poder proponer mejoras en el código.

Podemos concluir que la tecnología usada para abordar el desarrollo de la interfaz gráfica es la biblioteca Qt y, en concreto, el binding PySide que usa el lenguaje Python. Finalmente, para poder ejecutar las funciones que están implementadas en el script ULLRToolbox nos ayudamos de RPy2 que permite obtener la salida tal y como se nos mostraría en el mismo R.

CAPÍTULO 5

Detalles de implementación

Una vez establecidas las herramientas y técnicas a usar para el desarrollo del Trabajo de Fin de Grado, se ha definido la estructura del sistema de archivos del proyecto.

Tal y como se ve en la siguiente imagen, la raíz del proyecto cuenta con seis carpetas principales, tres archivos Python y uno de definición de recursos. La carpeta *build* contiene el proyecto construido para los diferentes sistemas operativos. Por ahora contiene dos subcarpetas que contiene los archivos ejecutables para linux de 32 bits y para 64 bits, aunque también puede construirse para Windows u otros sistemas modificando el archivo setup.py del que hablaremos más adelante.

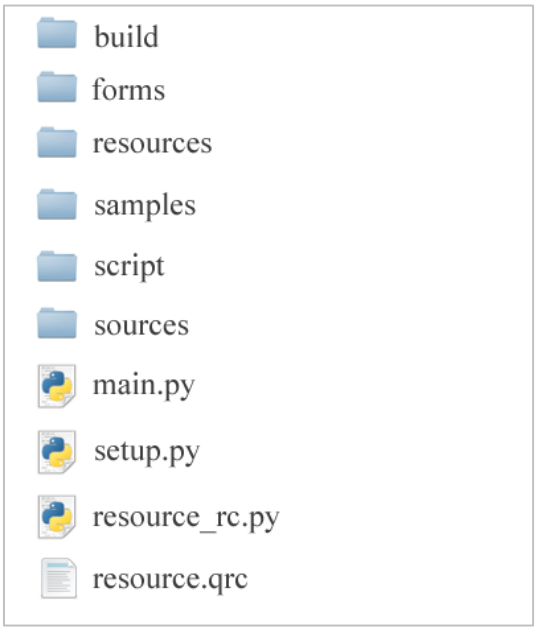


Figura 1. Estructura de archivos del proyecto

La carpeta *forms* contiene todos las interfaces creadas con el Qt Designer. Una lista de 29 archivos entre los cuales se incluye la ventana del proyecto (mainwindow.ui), mientras que el resto son cada uno de los diálogos asociados a cada función del programa.

La carpeta *resources* contiene todas las imágenes que se usan en el programa. Estas imágenes son, básicamente, el icono del programa y los iconos de la barra de tareas.

En la carpeta *samples*, tal y como su nombre indica, se encuentran una lista de ejemplos de archivos para que el usuario pueda probar las funciones disponibles en la documentación de ULLRToolbox sin necesidad de estar descargándoselos.

En la carpeta *script* se encuentra el archivo escrito en R donde están programadas todas las funciones que componen la caja de herramientas creadas por los desarrolladores de ULLRToolbox. Esta carpeta permite que si los programadores del toolbox actualizan su

script, en cualquier momento el usuario puede descargárselo e incorporarlo a esta carpeta sin necesidad de descargarse de nuevo el GUI. Automáticamente, al iniciar el programa gráfico este hace una comprobación y pide al usuario que instale, si las hay, las nuevas librerías.

Por último, la carpeta *sources* contiene todos los ficheros Python que permiten darle funcionalidad a la interfaz gráfica del proyecto, tanto los diálogos como la ventana principal.

Una vez definida la estructura de archivos, hay que interconectar los archivos que definen la interfaz gráfica (a partir de ahora denominados ui) con el código fuente. En PySide, para llamar a estos ficheros es necesario utilizar la librería llamada QUiLoader, que permite cargar los archivos ui desde el código python. La política adoptada para cargar los ficheros ui, es crear un archivo de recursos (llamados archivo qrc) que contiene la definición de todos los recursos a utilizar en el código, tales como los archivos ui o las imágenes que se encuentran en la carpeta resources. Esto permite que a estos archivos se les pueda dar un nombre simbólico y no necesita llamar a estos archivos usando una ruta absoluta. Una vez incorporado todos los ui al fichero qrc hay que ejecutar un comando desde la terminal de Linux que permite traducir este archivo qrc a uno de Python. Estos archivos son los que se encuentran en la figura 1 denominados *resource.qrc* y *resources_rc.py*.

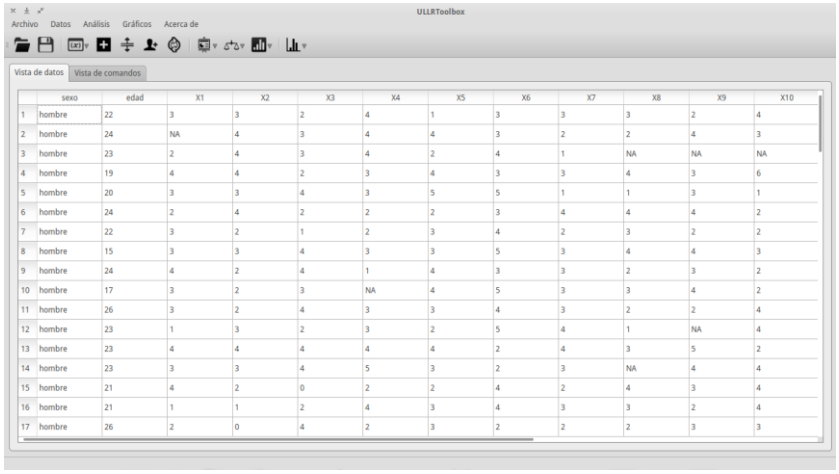
Otro archivo que se encuentra en la raíz del proyecto es el *main.py*. Este fichero python se encarga de cargar todos los archivos ui y pasarlos como parámetro cada uno de estos a cada una de las clases que se encuentra en el directorio sources. También se encarga de ejecutar la ventana principal maximizada del proyecto y de establecer el icono de la aplicación.

Cabe destacar que se ha creado un icono propio (Figura 2) para la aplicación gráfica ya que el que se encontraba ya hecho en la documentación no podía establecerse debido a que no guardaba la debida relación de aspecto que debe tener un icono.

Figura 2. A izquierda, el logo de ULLRToolbox, a derecha, el icono diseñado para el GUI.

Una clase destacada dentro de la carpeta sources es la llamada *MainWindow.py*. Este fichero es importante porque representa la ejecución de la ventana principal que en el diseño también se llama mainwindow.ui. Se le denomina así de forma estándar en Qt independientemente del lenguaje a usar.

En la clase MainWindow se han programado todas las funcionalidades que va a tener nuestra ventana principal. Tal y como vemos en la *Figura 3 y 4*, la ventana principal se compone de un menú principal, un barra de herramientas con los accesos directos más usados, y dos pestañas con distintas funcionalidades.



	sexo	edad	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10
1	hombre	22	3	3	2	4	1	3	3	3	2	4
2	hombre	24	NA	4	3	4	4	3	2	2	4	3
3	hombre	23	2	4	3	4	2	4	1	NA	NA	NA
4	hombre	19	4	4	2	3	4	3	3	4	3	6
5	hombre	20	3	3	4	3	5	5	1	1	3	1
6	hombre	24	2	4	2	2	2	3	4	4	4	2
7	hombre	22	3	2	1	2	3	4	2	3	2	2
8	hombre	15	3	3	4	3	3	5	3	4	4	3
9	hombre	24	4	2	4	1	4	3	3	2	3	2
10	hombre	17	3	2	3	NA	4	5	3	3	4	2
11	hombre	26	3	2	4	3	3	4	3	2	2	4
12	hombre	23	1	3	2	3	2	5	4	1	NA	4
13	hombre	23	4	4	4	4	4	2	4	3	5	2
14	hombre	23	3	3	4	5	3	2	3	NA	4	4
15	hombre	21	4	2	0	2	2	4	2	4	3	4
16	hombre	21	1	1	2	4	3	4	3	3	2	4
17	hombre	26	2	0	4	2	3	2	2	2	3	3

Figura 3. Pestaña de la vista de datos.

En la vista de datos toda la ventana principal se centra en mostrar una tabla en la que las columnas son las variables y las filas los sujetos. Esta tabla se carga al seleccionar un archivo externo mediante la función *lee_archivo* del toolbox. Esta vista permite que el usuario pueda visualizar de forma constante su data set de trabajo. El componente usado para implementar esta funcionalidad es el llamado *QTableWidget* que permite de forma sencilla insertar elementos y detectar si estos han cambiado.

El programa tiene una limitación y es que no puedes visualizar dos datasets desde la interfaz de datos. El programa asume que el dataset principal es el denominado ‘datos’ por lo que todo en el software gira en torno a esta variable. Es posible definir otros datasets por vía de comandos y visualizarlos pero no se podrá manipular ni usar las funciones con un dataset que no sea el definido como ‘datos’. En posteriores versiones podría ser de gran utilidad añadir esta funcionalidad aunque requeriría de cierta complejidad a nivel de desarrollo.

En la otra pestaña del programa llamada ‘vista de comandos’ disponemos de tres partes principales, la lista de variables, ventana de comandos y ventana de resultados.

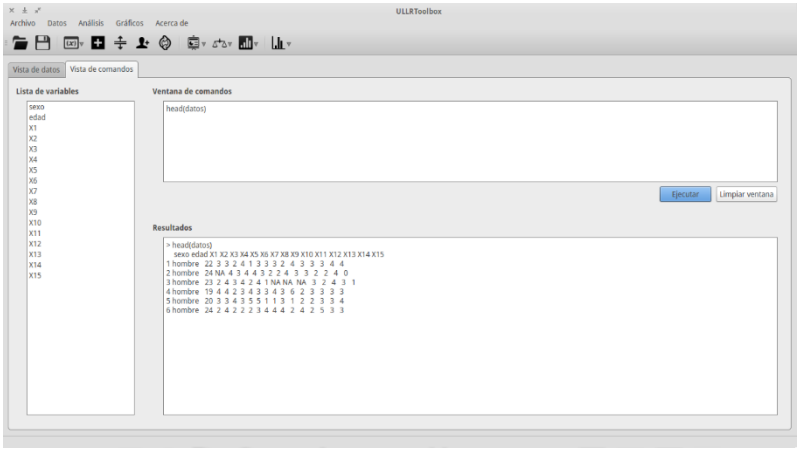


Figura 4. Pestaña de la vista de comandos.

En la lista de variables se nos mostrará un listado con los parámetros que tiene el dataset principal. Esto tiene su utilidad para ver en todo momento cuales son las variables a manipular sin necesidad de mostrarlas mediante la línea de comandos.

En la ventana de comandos, tal y como su nombre indica, podemos escribir código R a mano o llamar a las funciones del script. Esta ventana tiene una particularidad y es que, por un lado, podemos ejecutar el comando pulsando el botón de ejecutar, sin embargo, esto no es usable ya que tenemos que alternar entre teclado y ratón simultáneamente. Sin embargo, se ha creado un ‘listener’ o ‘evento’ que permite escuchar si estando en la ventana de comandos pulsamos la tecla ‘enter’. Si esto sucede, ejecuta el comando. Para poder hacerlo, fue necesario sobrescribir el método *eventFilter* de la clase *QObject*.

Por último, tenemos la ventana de resultados, en la que podemos visualizar la entrada y la salida de un comando realizado bien en la ventana de comandos o como resultado de la ejecución de un diálogo del GUI. La solución adoptada para implementar esta ventana ha sido la de un QTextEdit, que es un widget que permite insertar texto que también ha sido usado en la ventana de comandos. Sin embargo, en la ventana de resultados se le ha asignado el permiso de solo lectura por lo que el usuario no puede modificar los resultados pero si seleccionar el texto y copiarlo. También, se ha incorporado un botón que permite limpiar la ventana de los resultados y comandos ejecutados.

A la hora de ejecutar los comandos se ha usado un binding de R para python llamado *RPy2*. Lo que se intentó hacer desde un principio era poder ejecutar un comando de R y que obtuviéramos la salida tal y como nos la mostraría si ejecutásemos un comando en R. Para solventar esto, se utilizaron los '*callbacks'* que proporciona RPy2 a bajo nivel. Se creó una función a la que se le pasa una cadena de texto con el comando deseado, este se ejecuta mediante las funciones de alto nivel de RPy y la salida se muestra en la ventana de resultados haciendo uso de la función '*write_console'* de bajo nivel.

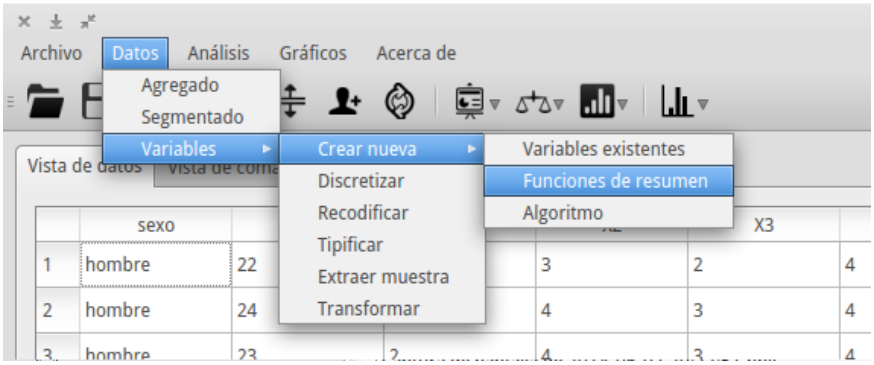


Figura 5. Barra de menú con las opciones principales.

En la parte superior de la ventana principal tenemos el menú con las funciones principales que encontramos en el script del toolbox (*Figura 5*). Desde esta barra podemos ir llamando a cada uno de los diálogos que permiten ejecutar los comandos de R de forma interna. En la parte inferior de este menú tenemos una barra de herramientas que básicamente son enlaces a las partes más relevantes de opciones que se encuentran en el menú. Los iconos han sido seleccionados bajo licencia *Creative Commons* (Atribution-Noncommercial) por lo que permite su uso con fines no comerciales.

Todas las opciones del menú provoca el despliegue de un dialogo en el que podemos seleccionar de forma gráfica las distintas opciones. Sin embargo, en el apartado 'Archivo' hay tres opciones que son abrir archivo, guardar archivo y exportar que hacen uso de un tipo de dialogo que ya está definido de forma nativa en Qt llamado QFileDialog. Esto facilita mucho la selección del directorio donde se quiere abrir o guardar un archivo (*Figura 6*).

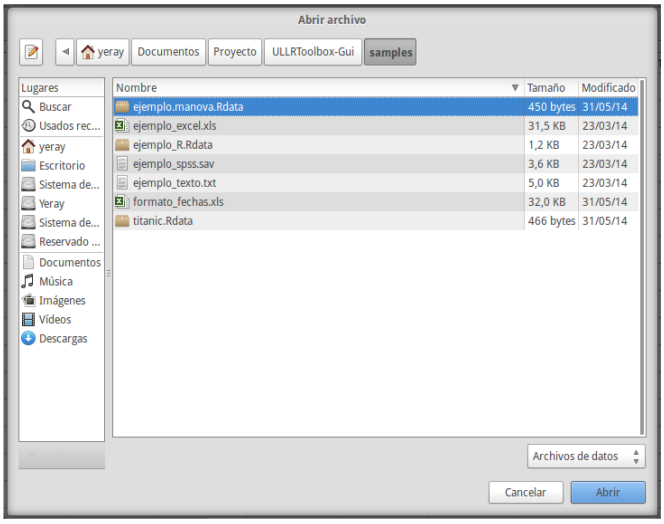


Figura 6. Dialogo de tipo QFileDialog que permite abrir un archivo en los formatos permitidos.

El resto de los diálogos han sido diseñados usando los distintos widgets que nos proporciona Qt. Los widgets usados en el desarrollo de estos diálogos son los reflejados en la Tabla 2.

QComboBox	QLineEdit	QPushButton
QTableWidget	QTextEdit	Spacer
QListWidget	QLabel	QTextBrowser
QRadioButton	QGridLayout	Line
QCheckBox	QGroupBox	QTableWidget

Tabla 2. Listado de widgets usados en el proyecto.

Así mismo, se ha desarrollado un widget para poder seleccionar distintas variables, de forma similar al que podemos encontrar en el programa de análisis estadístico SPSS (*Ver figura 7 derecha*).

Una vez que el usuario ha elegido todas las opciones de un dialogo y pulsa el botón aceptar, se hace una comprobación de que los campos han sido correctamente seleccionados y se construye el comando o función de ULLRToolbox y se le pasa al método que se encarga de ejecutar el comando y mostrarlo en la ventana de resultados. Todos los diálogos siguen la misma mecánica pero con distintos widgets y distintas

condiciones. En el caso de una mala selección de las opciones puede salir error en la ventana de resultados o si los parámetros obligatorios no han sido seleccionados no se ejecuta el comando y vuelve a salir el dialogo. Podemos ver un ejemplo de dos diálogos en la figura 7.

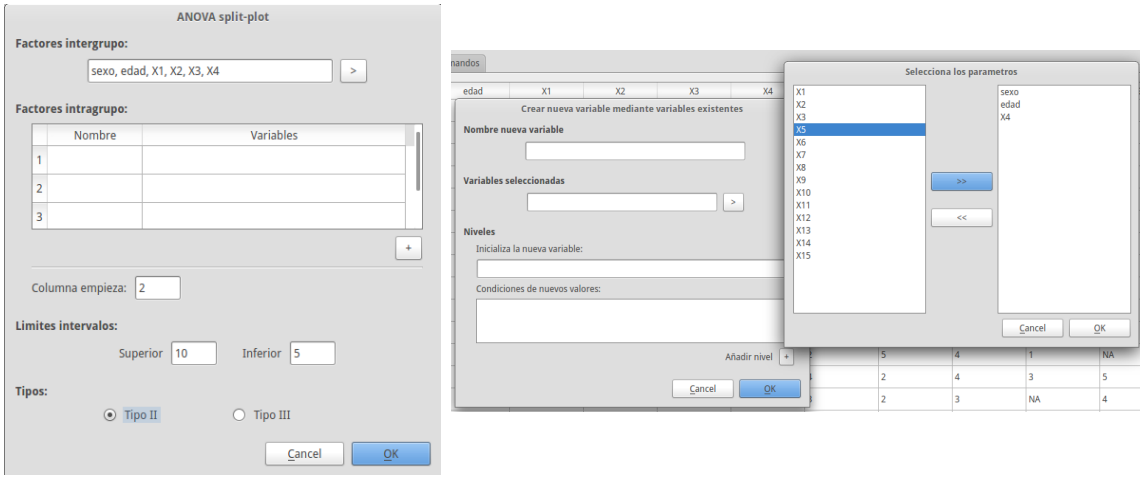


Figura 7. A la izquierda, el dialogo que permite hacer el ANOVA tipo Split-plot. A la derecha, crear una nueva variable mediante variables existentes.

Finalmente, el programa también permite visualizar gráficas (*Figura 8*). Estás gráficas se abren en una nueva ventana que no es propia del GUI sino que se abre en una ventana de R por lo que dota a las gráficas de la calidad propia del software estadístico.

Sin embargo, surgió un problema debido a que estas ventanas no podían cerrarse ni manipularse ya que no se estaba monitoreando los eventos de R. Para solucionar esto se tuvo que crear un proceso en segundo plano usando la función QTimer de Qt para que cada 20 milisegundos un método procesase los eventos que se producían en la interfaz de R.

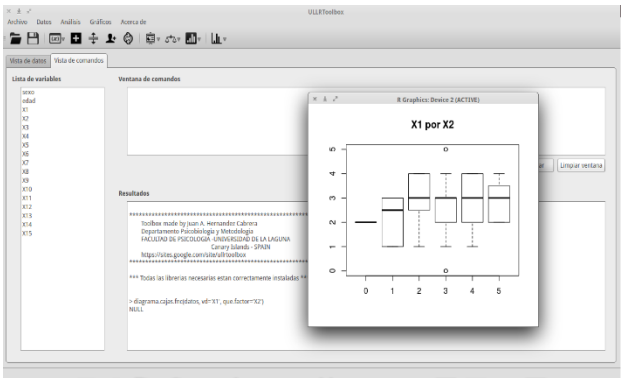


Figura 8. Diagrama de cajas en una nueva

CAPÍTULO 6

Conclusiones

Para cerrar este Trabajo de Fin de Grado, presentamos las conclusiones extraídas y derivadas del análisis y desarrollo del mismo.

Tal y como comentamos en la introducción, desde la creación del script ULLRToolbox se trató de lograr que R tuviera un uso más sencillo de cara al usuario común aliviándolo de la elevada curva de aprendizaje que supone aprender la compleja sintaxis de este lenguaje. El objetivo de este proyecto ha sido dar una vuelta de tuerca más y facilitar al usuario una interfaz gráfica que proporcione una mejor usabilidad a la hora de realizar el análisis estadístico, basándonos en la interfaz del potente software estadístico conocido como SPSS.

De este modo, tras elegir y aprender a usar las herramientas necesarias para comenzar a desarrollar un entorno gráfico, hacer un análisis de requisitos del mismo y familiarizarse con el script de ULLRToolbox, hemos logrado crear una interfaz gráfica usable y sencilla que permite que el usuario pueda ejecutar distintas operaciones estadísticas de una manera sencilla y accesible sin que tenga que estar familiarizado con la programación en el lenguaje R.

Como se ha detallado en apartados anteriores, la interfaz gráfica del nuevo programa contiene dos pestañas en la ventana principal, una para poder visualizar el dataset y otra para poder ver los resultados de las operaciones, englobadas por un menú en el cual el usuario puede seleccionar aquella función o método estadístico que desee (correlación, anova, estadísticos descriptivos, etc.) y en el cuadro de dialogo podrá seleccionar e introducir los datos necesarios para realizar el análisis. Posteriormente, el programa devolverá el resultado de esta operación en la vista de comandos o, si se trata de una gráfica, en una nueva ventana.

La creación múltiples espacios de trabajo totalmente independientes en la ventana principal permite que el usuario no tenga que seguir una línea uniforme de trabajo como puede ser una con ventana de comandos, sino que a la vez que el usuario puede ejecutar un comando, también puede visualizar en el panel izquierdo las variables disponibles.

Mientras en otra pestaña puede consultar en cualquier momento el estado de su dataset sin necesidad de ejecutar un comando para visualizarlo.

Todo esto ha permitido que se haya logrado desarrollar una interfaz gráfica que permite que ULLRToolbox esté totalmente integrado en la misma por lo que el usuario solo debe tener R instalado y luego debe instalar el GUI para poder empezar a hacer sus análisis estadísticos.

Sin embargo, existen algunos aspectos que se pueden mejorar para hacer que ULLRToolbox GUI sea una alternativa en el software de análisis estadístico:

- La opción de manipular varios datasets. Hasta el momento solo se puede visualizar uno simultáneamente por lo que las operaciones se tienen que realizar sobre este. Una posible mejora sería la posibilidad de poder visualizar varios datasets como puede hacer un programa algo distinto que es el excel y a la hora de realizar las operaciones seleccionar el tipo de dataset sobre el que quieres realizar la acción.

- Introducción de las técnicas de análisis multivariado, que permiten la realización de análisis estadísticos mediante la manipulación de un conjunto de variables y en los que se estudie en mayor profundidad la relación existente entre cada una de ellas, con los correspondientes menús y cuadros de diálogos para orientar al usuario las operaciones que puede realizar al trabajar con estos datos.

- Dotar de más funcionalidades a la ventana de comandos como la implementación de un historial de comandos que recuerde los últimos que ha utilizado (tanto de forma manual o gráficamente). También podría implementarse coloreado de sintaxis en esta parte de la aplicación.

En conclusión, los objetivos alcanzados en este proyecto, junto con estas mejoras arriba propuestas, permitirán que ULLRToolbox avance aún más a la hora de realizar análisis estadísticos complejos y de mayor potencia. Por tanto, puede facilitar aún más la tarea del analista mediante la vista de menús y de variables, y la manipulación de datos por parte de los usuarios, llegando a convertirse así en un método de aprendizaje cómodo, sencillo y eficaz.

CAPÍTULO 7

Conclusions

To finish this Final Degree Project, we show the conclusions extracted from the analysis and development of this project.

According to the introduction, since the creation of the script ULLRToolbox, it tried to get that R had an easier use to common user, relieving him of the high learning curve that the complex syntax of this language involves to learn. The target of this project has been facilitated a GUI that provides a better usability when do the statistical analysis, basing on the GUI of the powerful statistic software, known as SPSS.

For this way, after to choose and learn to use the necessary tools to start to develop a GUI, to make an analysis of its requirements and to acquaint oneself with the script of ULLRToolbox, we have achieved to create an usable and easy GUI that allows user to execute different statistical operations, in a simple and accessible way without he needs to be acquainted on programming R.

As it has elaborated in previous headlands, GUI of the new program counts with two tabs in the main window, one of them to visualize dataset, and the other of them to see the results of the operations, encompassed by a menu in which user can select the function or statistical method that he wants (correlate, ANOVA, descriptive statistics etc.) and in the dialog he could select and introduce the necessary data to do the analysis. Later, the program gives the result of this operation back in the commands view or, if is the case of a graphic, in a new window.

The development of multiple work spaces, totally independent between them, in the main windows allows user do not have to go on an uniform work line as a commands view, but user can execute a command, at the same time he can see in the left panel the available variables. While in another tab, he can consult the state of his dataset without executing a command to visualize it.

All of this has allowed that a GUI were development, which allows ULLRToolbox is integrated totally in this GUI, so user only needs to have installed R and then, he has to install the GUI to start to do his statistical analysis.

However, there are some bearings that can improve to do ULLRToolbox GUI were an alternative in statistical analysis software:

- The option of manipulating several datasets. To right moment, you can only visualize a dataset at the same time, so the operations have to be doing with it. A possible improvement would be the possibility of visualize some datasets, as other different programs as excel, and you could choose one of them to work on it.
- The introduction of multivariate analysis techniques, that allows to do statistical analysis trough the manipulation of several variables, and in which you could learn deeper the possible relation between them, with the menus and dialogs to guide user about the operations he can make because of working with this data.
- To endow more functionality to commands view as the implementation of a command history to remember the last paths you have used (either manually or graphically). Also, it could be implemented syntax coloring in this part of the application.

To conclude, the targets achieved in this Project, together with the improvements proposed above, will allow ULLRToolbox advances further when performing complex statistical analysis and more power, and therefore to facilitate the task of the analyst even more trough menu and variables view, and manipulation of data by users, reaching thus become a convenient, simple and effective learning method.

BIBLIOGRAFÍA

[1] J. A. H. Cabrera, «ULLRToolbox,» [En línea]. Available: https://sites.google.com/site/ullrtoolbox/. [Último acceso: 05 06 2014].

[2] Wikipedia, «Qt Software,» [En línea]. Available: https://en.wikipedia.org/wiki/Qt_(software). [Último acceso: 06 06 2014].

[3] Wikipedia, «SPSS,» [En línea]. Available: https://es.wikipedia.org/wiki/SPSS. [Último acceso: 06 06 2014].

[4] L. G. &. r. contributors, «Rpy2,» [En línea]. Available: http://rpy.sourceforge.net/. [Último acceso: 07 06 2014].

[5] Qt Project, «Qt Pyside,» [En línea]. Available: http://qt-project.org/wiki/pyside. [Último acceso: 04 06 2014].

[6] PySide Comunity, «PySide: Python for Qt,» [En línea]. Available: https://srinikom.github.io/pyside-docs/index.html. [Último acceso: 04 06 2014].

[7] After Hours Programming, «Python Introduction,» [En línea]. Available: http://www.afterhoursprogramming.com/tutorial/Python/Introduction/. [Último acceso: 08 06 2014].