Módulo 4 - Aprendizaje Supervisado: Técnicas de Regresión.

4.1 Regresión - ¿Qué, para qué y cómo?

Autores:

Por Rafael Alcalá

Catedrático de la Universidad de Granada

Instituto Andaluz Interuniversitario en Data Science and Computational Intelligence (DaSCI)

Y Augusto Anguita-Ruiz

Investigador postdoctoral en Instituto de Salud Global de Barcelona- ISGlobal.

Recordatorio: Introducción a NoteBook

Dentro de este cuaderno (*NoteBook*), se le guiará paso a paso desde la carga de un conjunto de datos hasta el análisis descriptivo de su contenido.

El cuaderno de *Jupyter* (Python) es un enfoque que combina bloques de texto (como éste) junto con bloques o celdas de código. La gran ventaja de este tipo de celdas, es su interactividad, ya que pueden ser ejecutadas para comprobar los resultados directamente sobre las mismas. *Muy importante*: el orden las instrucciones es fundamental, por lo que cada celda de este cuaderno debe ser ejecutada secuencialmente. En caso de omitir alguna, puede que el programa lance un error, así que se deberá comenzar desde el principio en caso de duda.

Antes de nada:

Es muy muy importante que al comienzo se seleccione "*Abrir en modo de ensayo*" (draft mode), arriba a la izquierda. En caso contrario, no permitirá ejecutar ningún bloque de código, por cuestiones de seguridad. Cuando se ejecute el primero de los bloques, aparecerá el siguiente mensaje: "*Advertencia: Este cuaderno no lo ha creado Google.*". No se preocupe, deberá confiar en el contenido del cuaderno (*NoteBook*) y pulsar en "Ejecutar de todos modos".

¡Ánimo!

Haga clic en el botón "play" en la parte izquierda de cada celda de código. Las líneas que comienzan con un hashtag (#) son comentarios y no afectan a la ejecución del programa.

También puede pinchar sobre cada celda y hacer "*ctrl+enter*" (*cmd+enter* en Mac).

Cada vez que ejecute un bloque, verá la salida justo debajo del mismo. La información suele ser siempre la relativa a la última instrucción, junto con todos los `print()` (orden para imprimir) que haya en el código.

ÍNDICE

En este *notebook*:

1. Aprenderemos los conceptos generales de los métodos de regresión.
2. Plantearemos sus posibles aplicaciones bioinformáticas haciendo uso de conjuntos de datos biológicos.

Contenidos:

1. INTRODUCCIÓN A LAS TÉCNICAS DE REGRESIÓN Y SU RELEVANCIA EN EL CAMPO DE LA BIOINFORMÁTICA

Las tecnologías ómicas de alto rendimiento como los **microarrays de ADN**, la **espectofotometría de masas** o los ***chips* de proteínas** han sido ampliamente utilizados durante los últimos años en las **ciencias biomédicas** para desvelar los mecanismos moleculares que subyacen a la enfermedad. Entre otros parámetros, estas técnicas permiten conocer de una sola batida el nivel de expresión de miles o millones de **genes, metabolitos o proteínas** en una amplia variedad de tejidos biológicos. Los datos que se derivan de estas técnicas, conocidos en su conjunto como "**datos ómicos**", constituyen por lo tanto agrupaciones de una **alta complejidad y dimensionalidad**. Entre sus **principales características**, destaca el hecho de que presentan **dominios continuos** (i.e. la expresión de un gen/metabolito/proteína comprenderá valores de 0 en adelante), la existencia de **fenómenos de colinealidad** entre variables (i.e. dos genes participantes en la misma ruta celular tendrán niveles de expresión muy similares), así como la presencia en fenomenos de **interacción epistática** (a través de los cuales ciertos genes son capaces de modular el efecto que otros genes tienen sobre el fenotipo).

Es imprescindible, por lo tanto, encontrar los métodos analíticos más apropiados y capaces de lidiar con estas particularidades. En la mayoría de las ocasiones, un investigador bioinformático trabajando con datos ómicos estará interesado en **identificar relaciones funcionales** entre cada gen/metabolito/proteína y el fenotipo de interés (asociaciones), **reducir el número de genes/metabolitos/proteínas de partida** generando un conjunto de datos menos complejo, o **generar un modelo con buena habilidad predictiva** sobre el fenotipo de interés (variable de salida).

Para todo ello, **las técnicas de regresión** que se introducen en este módulo son especialmente apropiadas, y se han consolidado como uno de los abordajes *gold-standard* cuando se trabaja en bioinformática. Entre los principales motivos, destaca su **diseño orientado a trabajar con variables continuas**, su versatilidad para **identificar variables redundantes** o su capacidad para **estimar** valores desconocidos. Más allá de los datos ómicos, las **ténicas de regresión** tambien han demostrado una **gran utilidad** cuando se trabaja con **otras fuentes de datos biológicas** comunmente utilizadas en biomedicina, como son los datos derivados de **microoscopía** (en los que las imagenes son transformadas en valores numéricos), o las **medidas antropométricas, clínicas y bioquímicas** que se obtienen habitualmente durante un estudio caso-control.

Por definición, las técnicas de regresión estudian las relaciones existentes entre **variables de entrada** (variables independientes), normalmente denominadas $X_1, X_2, \ldots, X_p$, y una **variable de salida** (variable dependiente), denominada Y. De esta forma, un regresor es capaz de estimar el valor esperado que tomaría Y, dado un conjunto de valores para las variables X. Enlazando con la parte anterior del curso, los métodos de regresión vienen a ser un **tipo de aprendizaje automático supervisado** cuando lo que se persigue es predecir un **valor continuo**. Para ello, se construyen modelos que especifiquen el cambio que produce en la respuesta (salida) cada variable de entrada, suponiendo que el resto permanecen constantes:

$$Y_i = f(X_i) + \epsilon_i$$

Donde f es una **función desconocida** y ϵ un **error aleatorio** con media en cero. A mayor varianza de ϵ mayor dificultad a la hora de estimar f. Dicha varianza suele ser más alta cuanto menor es el número de instancias de las que se dispone y/o mayor es el número de variables de entrada que realmente se necesitan, lo cual ocurre con bastante frecuencia en el campo de la bioinformática.

Un modelo de regresión puede tener muchas **formas distintas**: puede ser una **función** (vease la Figura 1), un **conjunto de reglas**, una **red neuronal**, **una máquina de soporte vectorial** (SVM), etc. En el presente módulo, vamos a ver muy brevemente tres de los métodos de regresión más conocidos: **Regresión lineal simple y múltiple**, el método de k**-vecinos más cercanos (KNN)** y el **modelo de regresión M5**.

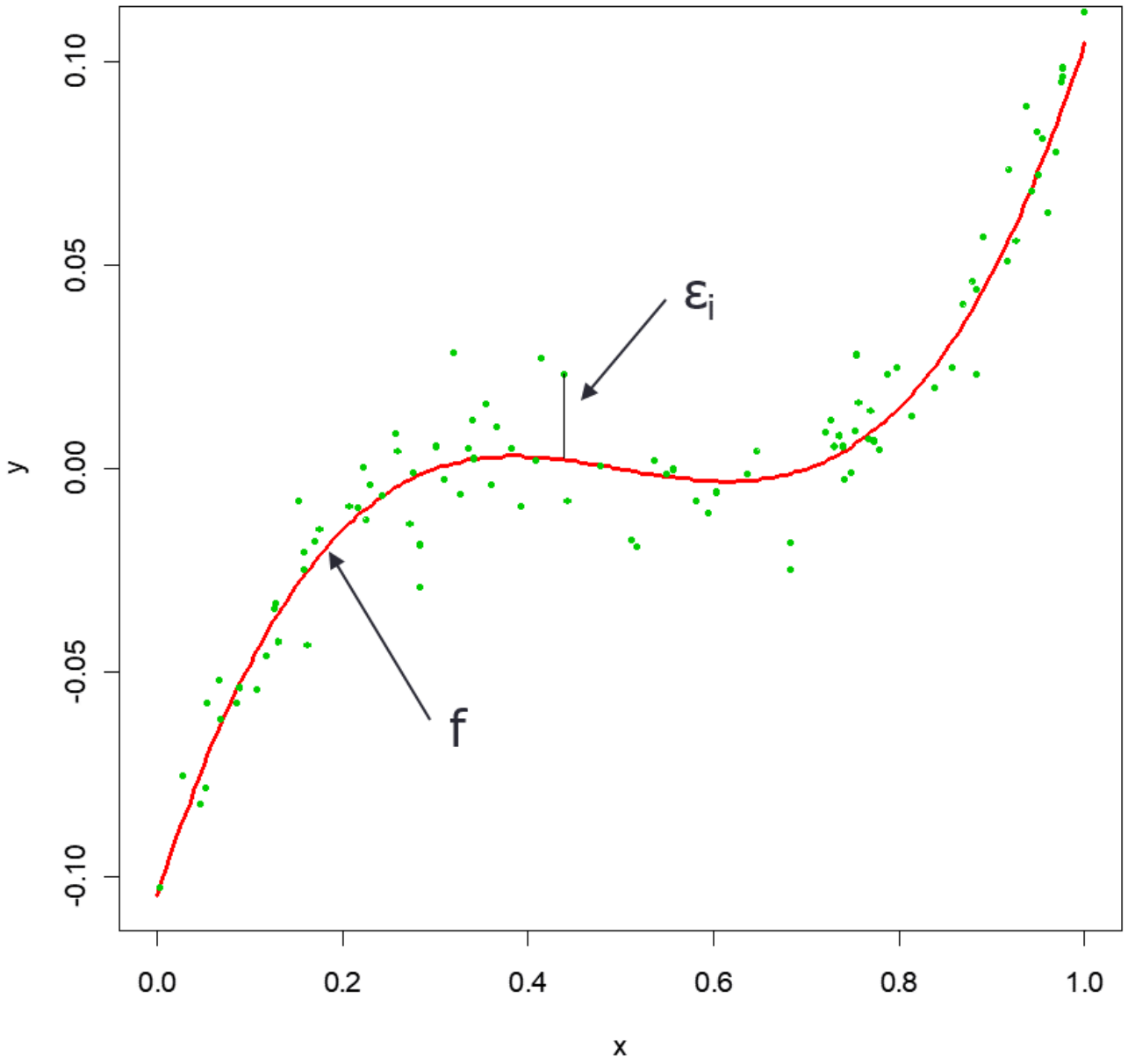

Figura 1. Ejemplo de regresión lineal simple: Estimación de la función f

2. MOTIVOS PARA QUERER TENER UNA BUENA ESTIMACIÓN DE F: EJEMPLOS SOBRE CONJUNTOS DE DATOS BIOLÓGICOS

En *Machine Learning*, y por lo tanto también en el caso de la regresión, se usan los datos para "aprender" f. Como hemos introducido a lo largo de esta cápsula, encontraremos dos razones principales por las que queremos obtener una buena estimación de f:

• **Predicción**: Si podemos producir una buena estimación para f (y la varianza del ϵ no es demasiado grande) podemos **hacer predicciones precisas** para la variable de salida, Y, basadas en un nuevo valor de X. Se intentan aprender modelos tan precisos como sea posible en este caso. Un ejemplo muy claro de este tipo de aplicación en bioinformática es el uso de **puntajes de riesgo poligenéticos** (o *Genetic Risk Scores (GRS)*, como se les conoce por sus siglas en Inglés). Los puntajes de riesgo poligenéticos son sistemas de puntuación que nos permiten **evaluar el riesgo genético de un individuo a padecer cierta enfermedad o rasgo fenotípico**. Se calculan como la **suma ponderada del número de alelos de riesgo** que un individuo porta entre una batería de marcadores genéticos. Una vez calculado el puntaje de riesgo poligenético de cada individuo, es habitual construir un **modelo de regresión** para, con estos valores como variable de entrada X, predecir los valores Y_i desconocidos de un **fenotipo continuo de interés** (p.ej. tiempo de supervivencia de un paciente o niveles altos de un biomarcador en sangre). La gran utilidad de los puntajes de riesgo poligenéticos y su incorporación en modelos predictivos de regresión radica en su capacidad para predecir fenotipos continuos de interés con una gran **anticipación temporal** (ya que la genética de un individuo es un parámetro que puede ser evaluado desde el momento del nacimiento). La aplicación de modelos predictivos de regresión basados en *GRSs* para estimar el **índice de masa corporal (o estado de obesidad) futuro** en niños con sobrepeso ha sido una técnica ampliamente utilizada en los últimos años.

• **Interpretación/explicación/descubrimiento**: Alternativamente, también nos puede interesar **desvelar el tipo de relaciones existentes** entre la variable de salida Y y cada una de las variables de entrada $X's$, así como entre las propias variables $X's$. Para ello, se intentan aprender modelos que expliquen la variable Y a partir de las variables $X's$. Con esta finalidad, los modelos de regresión también son muy utilizados en bioinformática; especialmente en el campo de las ciencias ómicas. Algunos ejemplos destacables incluyen; 1) El **descubrimiento de las variantes genéticas** responsables del desarrollo de una enfermedad (en este caso, un fenotipo continuo de interés como pueden ser los niveles altos de colesterol en sangre), o 2) La **identificación de fenómenos de interacción ómica** tales como los existentes entre variantes genéticas y metabolitos (*mQTLs*), variantes genéticas y metilación del ADN (*meQTLs*), o metilación del ADN y metabolitos (*mCpG*).

Ambas razones son normalmente contradictorias, predominando normalmente una sobre la otra dependiendo de las necesidades concretas del problema.

3. MÉTRICAS DE CALIDAD DE LOS MODELOS DE REGRESIÓN

Un modelo de regresión bien ajustado da como resultado valores estimados/pronosticados cercanos a los valores de los datos observados para una variable de salida o **dependiente** Y, a partir de los valores observados en otras variables de entrada o **independientes** $X's$. Si no hubiera variables de entrada informativas, generalmente se usaría el modelo de la media, que utiliza directamente la media de los datos Y observados como predicción para cualquier nuevo valor desconocido. En el caso de buscar relaciones lineales entre las variables de entrada y la variable de salida, éste sería el peor de los escenarios. Por lo tanto, siempre se buscará que el ajuste de un modelo de regresión propuesto sea mejor que el ajuste del modelo de media.

En regresión, como en otras áreas del *Machine Learning*, **existen muchas métricas** para evaluar de la calidad de los modelos obtenidos: **Error Cuadrático Medio**, **Raíz del Error Cuadrático Medio (RECM)**, R^2 (cuando su cálculo es posible y/o tiene sentido), **Error Absoluto Medio**, etc. En esta sección hablaremos de los dos estadísticos más ampliamente utilizados entre los anteriores: el R^2 y la **RECM**. El cálculo de ambos parámetros se basa en dos sumas de cuadrados: **Suma total de los cuadrados** (o *total sum-of-squares TSS*) y **suma de cuadrados de los errores** (o *residual sum-of-squares RSS*). *TSS* mide qué tan lejos están los datos del peor modelo que se podría aprender mediante la técnica de aprendizaje utilizada (por ejemplo, como de lejos están los datos de una distribución de línea constante centrada en la media, que sería el peor modelo en el caso de un aprendizaje por regresión lineal multivariable). *TSS* no se puede calcular (o no tiene sentido hacerlo) para todos los métodos de regresión, por lo que las medidas basadas en *TSS* no siempre estarán disponibles. Por otro lado, *RSS* mide qué tan lejos están los datos de los valores pronosticados por el modelo aprendido (diferencia en valor absoluto).

Las diferentes combinaciones de estos dos valores proporcionan información diferente sobre cómo el modelo de regresión se compara con modelos obtenidos por una misma técnica o por distintas técnicas de regresión:

- R^2 **(o R-cuadrado)**: La diferencia entre *TSS* y *RSS* representa la mejora en la predicción del modelo de regresión, en comparación con el peor modelo que teóricamente se podría obtener con dicha técnica. Dividir esa diferencia por *SST* da R^2. Por lo tanto, R^2 es la mejora proporcional en la predicción del modelo de regresión, en comparación con el peor modelo y nos indica la bondad de ajuste del modelo para una técnica en concreto. R^2 tiene la propiedad útil de que su escala es intuitiva: varía de cero a uno, con 0 indicando que el modelo propuesto no mejora la predicción sobre el peor modelo teórico de dicha técnica, y 1 indicando una predicción perfecta. La mejora en el modelo de regresión resulta en aumentos proporcionales de R^2.

$$R^2 = (TSS - RSS)/TSS = 1 - RSS/TSS$$

- **RECM (*Root Mean Squared Error -RMSE-* en inglés)**: La raíz del error cuadrático medio es la raíz cuadrada de la varianza de los residuos. En este caso, sólo se usa *RSS* (diferencias al cuadrado), por lo que es aplicable a los modelos obtenidos por cualquier técnica de regresión. RECM indica el ajuste absoluto del modelo a los datos: cuán cerca están los puntos de datos observados de los valores pronosticados por el modelo. Siendo n el número de datos y e_i cada una de las diferencias entre dato observado y valor pronosticado por el modelo ($e_i = y_i - y_i'$), RECM se puede calcular como:

$$RECM = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2}$$

Mientras que R^2 es una medida relativa de ajuste, **RECM** es una medida absoluta. Por lo tanto, tiene la propiedad útil de estar en las mismas unidades que la variable de respuesta. Los valores más bajos de **RECM** indican un mejor ajuste. **RECM** es una buena medida de la precisión con la que el modelo predice la respuesta, y es el criterio más utilizado para medir el ajuste si el propósito principal del modelo es la predicción.

4. DESCRIPCIÓN DEL CONJUNTO DE DATOS A UTILIZAR EN ESTE MÓDULO

Siempre que hablemos de variables **continuas**, variables **enteras** o variables **categóricas en las que la información pueda ser representada de manera numérica siguiendo un orden lógico** (también conocidas como variables **ordinales**), los métodos de regresión serán de **aplicación y utilidad**. Este punto es sumamente importante y es la principal diferencia entre los métodos de regresión y clasificación. En clasificación tendrán sentido las variables categóricas no ordinales. Sin embargo, en regresión no tienen cabida.

Sobre el conjunto de datos con el que se ha venido trabajando desde el inicio del curso, ***The Cancer Genome Atlas (TCGA)***, podemos destacar varios escenarios en los que la aplicación de métodos de regresión sería de interés. Por ejemplo, podríamos utilizar un modelo de regresión para predecir el **tiempo de superviciencia** de los pacientes con melanoma, basándonos en la expresión de aquellos genes que previamente han demostrado una implicación directa en la fisiopatología de la enfermedad. Por otro lado, podríamos utilizar modelos de regresión para **conocer relaciones entre variables (en este caso genes)** y desvelar fenómenos de interacción molecular, por los que los productos de unos u otros genes interactuan entre sí para condicionar la aparición o empeoramiento de la patología.

Aunque *a priori* ambas son hípotesis válidas, existe un problema cuando queremos inferir un parámetro tan determinante como es la **supervivencia del paciente**, utilizando unicamente datos de **caracter genético**. Y es que la expresión de un gen es una única y pequeña pieza dentro del complejo mecanismo que subyace la aparición y malignización de un tumor. Por otro lado, aunque en el conjunto de datos original encontramos **variables clínicas** alternativas que podrían ser de utilidad, la mayoría de ellas **no presentan un dominio continuo**.

Tomando esto en consideración, en este módulo, plantearemos un **conjunto de datos alternativo** conformado por datos clínicos de **obesidad infantil** obtenidos en pacientes que podrían presentar **insulino-resistencia**. Este tipo de datos, que incluyen tanto datos clínicos como antropométricos, si que han demostrado una gran utilidad para predecir una variable de salida de tal relevancia clínica como es el índice de insulino-resistencia (***HOMA-IR* por sus siglas en Inglés**). Entre otras aplicaciones, la estimación del índice *HOMA-IR* en niños con sobrepeso u obesidad nos permite identificar niños en alto riesgo de desarrollar *Diabetes Tipo 2* en el futuro.

Este nuevo conjunto de datos de obesidad contiene información para **16 variables** en **292 niños**. Dicha información se deriva de una cohorte de niños españoles con edades comprendidas entre 5 a 15 años, agrupados en tres condiciones experimentales (normopeso, sobrepeso y niños con obesidad). Para ellos, hay disponible una amplia gama de datos clínicos, bioquímicos y antropométricos (composición corporal), así como datos de estilo de vida y actividad física (obtenidos mediante acelerómetros). A través de la aplicación de **técnicas de regresión** sobre este conjunto de datos, se podría estudiar **la relación entre las variables de antropometría, bioquímica, actividad física y el estado de insulino-resistencia**, así como **predecir el propio índice de insulino-resistencia *HOMA-IR* a partir de las anteriores**. Para poder mostrar el funcionamiento de las técnicas de regresión presentadas en este móludo sólo se considerará una **submuestra representativa de la población mencionada**. Dicha **submuestra consta de 292 individuos** que a su vez presentan datos de actividad física de calidad suficiente.

En los análisis que abordaremos en este módulo, **el índice de resistencia a la insulina *HOMA-IR* se incluirá como variable de salida**, constituyendo el **output** de nuestros modelos de regresión, ya que se ha validado ampliamente como un buen indicador del estado de insulino-resistencia y pre-Diabetes en niños y adultos. Como variables de entrada en el modelo, emplearemos datos clínicos y antropométricos (como el *sexo*, la *edad*, el *estadío puberal*, la *altura*, la *circunferencia de cintura* y el *Índice de Masa Corporal (IMC)*), principales mediciones de actividad física (tales como el número de *minutos diarios de inactividad (sedentarismo)*, o de *actividad física leve, moderada y vigorosa*). Además, se incluirán indicadores de disfunción cardio-metabólica distintos al propio *HOMA-IR* (como son los niveles de *colesterol HDLc y LDLc*, *triglicéridos* y la presión sanguinea). Todas ellas están

5. ¿POR QUÉ USAR R? INSTALACIÓN DEL ENTORNO R Y SUS BIBLIOTECAS PARA REGRESIÓN

Al igual que vimos en el Módulo 2 para la carga de datos, y aunque *Python* es el lenguaje vehicular de este curso, las bibliotecas de *R* nos ofrecen una manera **mucho más cómoda** de aplicar los algoritmos de regresión que veremos en éste módulo. En *R*, los algoritmos permiten realizar una formulación directa sobre el uso de las variables de cualquier conjunto de datos. Ésto permite realizar una construcción por pasos de los modelos, integrando/determinando las variables realmente relevantes y descubriendo el tipo de relación/interacción existente entre las mismas. Igualmente, en *R* se dispone de más algoritmos y, en nuestro caso, se hace necesario para poder utilizar el algoritmo M5 (vía *Cubist*) por ser casi un paradigma en el área de la regresión (véase el estudio realizado en 2019 entre 164 algorimos por *Gacto et. al.* donde un simple árbol se muestra competitivo contra los mejores *ensembles* de 500 árboles - *Sección de Bibliografía*). De este modo, en este módulo utilizaremos **funciones de *R* para aplicar los algoritmos de regresión** indicados.

La instalación de *R* en nuestro entorno de *Google Colab* ya se explicó en el Módulo 2 de este *MOOC*, por lo que aquí directamente incluimos las instrucciones pertinentes. Hay que recordar que todas las instalaciones de bibliotecas que realicemos en el entorno de *Google Colab* solo permanecerán activas unas pocas horas, después de las cuales las bibliotecas instaladas se eliminan. **Por lo tanto, será necesario que vuelvas a ejecutar los códigos de instalación de bibliotecas de esta sección cuando necesites ejecutar *notebooks* que contengan código de *R***. En cualquier caso, los códigos para realizar esta instalación estarán incluidos al inicio de cada una de las cápsulas en las que sea necesario.

Por último, recordemos que cuando una celda utiliza código de *R* esto se indica mediante la simbología **%%R** en los *notebooks* de *Python*. Quitándo dicha línea, que sirve para indicar que lo que sigue en el *notebook* está escrito en *R*, podríamos ejecutar las mismas instrucciones en cualquier entorno de *R* distinto al *notebook* sin problemas. Por ejemplo, en nuestra propia máquina *Windows, Linux, MacOS* utilizando nuestra instalación local de *R* y *RStudio*.

El siguiente trozo de código, correspondiente a la instalación de *R*, es exclusivo al uso de los *notebooks* de *Colab* y sería equivalente a instalar *R* y *RStudio* en nuestras máquinas:

In [1]:
```
# Tiempo estimado de ejecución: 20 segundos aprox.

### Instalación de R en notebooks de Google Colab ###
!apt-get update
!apt-get install r-base
!pip install rpy2==3.5.1
%load_ext rpy2.ipython
print ("Instalación de R en Google Colab terminada")
```

```
Get:1 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic InRelease [1
5.9 kB]
Hit:2 http://archive.ubuntu.com/ubuntu bionic InRelease
Get:3 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:4 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:5 https://cloud.r-project.org/bin/linux/ubuntu bionic-cran40/ InRelease [3,
626 B]
Hit:6 http://ppa.launchpad.net/cran/libgit2/ubuntu bionic InRelease
Get:7 http://ppa.launchpad.net/deadsnakes/ppa/ubuntu bionic InRelease [15.9 kB]
Get:8 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:9 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic InRelease [2
1.3 kB]
Ign:10 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_
64  InRelease
Ign:11 https://developer.download.nvidia.com/compute/machine-learning/repos/ubu
ntu1804/x86_64  InRelease
Get:12 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_
64  Release [697 B]
Get:13 https://developer.download.nvidia.com/compute/machine-learning/repos/ubu
ntu1804/x86_64  Release [564 B]
Get:14 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_
64  Release.gpg [836 B]
Get:15 https://developer.download.nvidia.com/compute/machine-learning/repos/ubu
ntu1804/x86_64  Release.gpg [833 B]
Get:16 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Package
s [1,396 kB]
Get:17 http://security.ubuntu.com/ubuntu bionic-security/restricted amd64 Packa
ges [324 kB]
Get:18 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packages
[1,963 kB]
Get:19 http://security.ubuntu.com/ubuntu bionic-security/multiverse amd64 Packa
ges [24.5 kB]
Get:20 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic/main Sources
[1,745 kB]
Get:21 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic/main amd64 P
ackages [893 kB]
Get:22 http://archive.ubuntu.com/ubuntu bionic-updates/restricted amd64 Package
s [353 kB]
Get:23 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages
[2,163 kB]
Get:24 http://archive.ubuntu.com/ubuntu bionic-updates/multiverse amd64 Package
s [31.4 kB]
Get:25 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [2,3
94 kB]
Get:26 http://ppa.launchpad.net/deadsnakes/ppa/ubuntu bionic/main amd64 Package
s [39.5 kB]
Get:27 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic/main amd64 P
ackages [49.4 kB]
Ign:28 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_
64  Packages
Get:28 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_
64  Packages [577 kB]
Get:29 https://developer.download.nvidia.com/compute/machine-learning/repos/ubu
ntu1804/x86_64  Packages [73.8 kB]
Fetched 12.3 MB in 3s (4,675 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
r-base is already the newest version (4.0.4-1.1804.0).
0 upgraded, 0 newly installed, 0 to remove and 60 not upgraded.
Requirement already satisfied: rpy2 in /usr/local/lib/python3.7/dist-packages
(3.4.2)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.7/dist-packages
```

```
(from rpy2) (2.11.3)
Requirement already satisfied: pytz in /usr/local/lib/python3.7/dist-packages
(from rpy2) (2018.9)
Requirement already satisfied: tzlocal in /usr/local/lib/python3.7/dist-package
s (from rpy2) (1.5.1)
Requirement already satisfied: cffi>=1.10.0 in /usr/local/lib/python3.7/dist-pa
ckages (from rpy2) (1.14.5)
Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.7/dis
t-packages (from jinja2->rpy2) (1.1.1)
Requirement already satisfied: pycparser in /usr/local/lib/python3.7/dist-packa
ges (from cffi>=1.10.0->rpy2) (2.20)
Instalación de R en Google Colab terminada
```

A partir de aquí, quitando la línea del comando **%%R**, es código *R* y podría ejecutarse, como se ha indicado, en una instalación local vía *RStudio*. El siguiente trozo de código se corresponde con la instalación e importación de las bibliotecas necesarias para el uso de los algoritmos de regresión indicados.

In [2]:
```
# Tiempo estimado de ejecución: 2:30 minutos aprox.
# Bibliotecas necesarias:
# ISLR para regresión lineal multivariable
# kknn para k-vecinos más cercanos de regresión
# Cubist para modelos de regresión basados en M5

%%R
### Instalación de las bibliotecas necesarias
install.packages(c("ISLR", "kknn", "Cubist"))
print ("Instalación de las bibliotecas de R para este módulo terminada")

### Importación de las bibliotecas necesarias ###
require(ISLR)
require(kknn)
require(Cubist)
print ("Importación de las bibliotecas de R para este módulo terminada")
```

```
R[write to console]: Installing packages into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

R[write to console]: also installing the dependencies ‘plyr’, ‘igraph’, ‘reshap
e2’


R[write to console]: trying URL 'https://cran.rstudio.com/src/contrib/plyr_1.8.
6.tar.gz'

R[write to console]: Content type 'application/x-gzip'
R[write to console]:  length 401191 bytes (391 KB)

R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]:
```

```
R[write to console]: downloaded 391 KB


R[write to console]: trying URL 'https://cran.rstudio.com/src/contrib/igraph_1.
2.6.tar.gz'

R[write to console]: Content type 'application/x-gzip'
R[write to console]:  length 2715500 bytes (2.6 MB)

R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]:

R[write to console]: downloaded 2.6 MB
```

```
R[write to console]: trying URL 'https://cran.rstudio.com/src/contrib/reshape2_
1.4.4.tar.gz'

R[write to console]: Content type 'application/x-gzip'
R[write to console]:  length 37307 bytes (36 KB)

R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]:

R[write to console]: downloaded 36 KB


R[write to console]: trying URL 'https://cran.rstudio.com/src/contrib/ISLR_1.2.
tar.gz'
```

```
R[write to console]: Content type 'application/x-gzip'
R[write to console]:  length 1757568 bytes (1.7 MB)

R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]:

R[write to console]: downloaded 1.7 MB


R[write to console]: trying URL 'https://cran.rstudio.com/src/contrib/kknn_1.3.
1.tar.gz'

R[write to console]: Content type 'application/x-gzip'
R[write to console]:  length 388410 bytes (379 KB)
```

```
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]:

R[write to console]: downloaded 379 KB


R[write to console]: trying URL 'https://cran.rstudio.com/src/contrib/Cubist_0.
2.3.tar.gz'

R[write to console]: Content type 'application/x-gzip'
R[write to console]:  length 463513 bytes (452 KB)

R[write to console]: =
R[write to console]: =
```

```
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]: =
R[write to console]:

R[write to console]: downloaded 452 KB


R[write to console]:

R[write to console]:
R[write to console]: The downloaded source packages are in
        ‘/tmp/RtmpyWoKR9/downloaded_packages’
R[write to console]:
R[write to console]:


[1] "Instalación de las bibliotecas de R para este módulo terminada"
```

```
R[write to console]: Loading required package: ISLR

R[write to console]: Loading required package: kknn

R[write to console]: Loading required package: Cubist

R[write to console]: Loading required package: lattice

[1] "Importación de las bibliotecas de R para este módulo terminada"
```

Importante: Si en algún momento durante el seguimiento de las cápsulas de este módulo le aparece el mensaje ***UsageError: Cell magic `%%R` not found*** tras ejecutar una celda de código, tendrá que ejecutar de nuevo la instalación de R y de las librerías explicadas en ésta sección.

6. CARGAR LOS DATOS

Una vez completada esta cápsula introductoria, en las siguientes profundizaremos en cada uno de los tres de los métodos de regresión más conocidos, aplicándolos sobre nuestro conjunto de datos de obesidad:

- La regresión lineal simple y/o múltiple
- El algoritmo de k vecinos más cercanos para regresión (KNN)
- Y el arbol de regresión M5 (vía *Cubist* en nuestro caso)

A continuación, se muestran las **líneas de código** necesarias para **cargar** el conjunto de datos de obesidad mencionado en el entorno de R; el cual va a ser utilizado en las siguientes cápsulas del módulo. Hay que tener en cuenta que el fichero que vamos a leer tiene **formato de tabla con cabecera**. En este caso, siendo ésto lo habitual, cada columna representa una variable del problema. La última columna del conjunto de datos se corresponde con la **variable de salida** que queremos inferir, es decir, el *output* de nuestro modelo de regresión. La primera línea del fichero de datos (cabecera de la tabla) incluye los nombres de todas las variables separadas por comas, correspondiendo cada nombre con cada una de las columnas anteriormente mencionadas. Igualmente, el resto de líneas del fichero (filas de la tabla) incluyen valores numéricos separados por comas (siendo de vital importancia que dicha información numérica vaya en formato inglés, porque los algoritmos que utilizaremos usan dicho formato y podrían confundir la coma de separación de datos con lo que sería la coma decimal en Español). A partir de la segunda línea del fichero, cada línea (fila de la tabla) contiene la información relativa a un niño. El tipo de fichero ideal para tal fin es el **CSV**, ya que puede ser fácilmente generado o editado desde *Microsoft Excel* u *OpenOffice Calc*.

Los datos se encuentran disponibles vía *Google Drive* en un fichero llamado ***homa.csv***. No obstante, si estuviésemos en local con *RStudio* y/o con otro fichero de datos, sólo tendríamos que cambiar *url("https://drive.google (https://drive.google)...")* por \"\<*Unidad*>:\\<*Ruta*>\\<*fichero*>\" (Ej: \"*C:\Datos\homa.csv*\" en el caso de un sistema operativo *Windows*). El método *head()* no es necesario pero sirve para comprobar que los datos se leyeron correctamente. Debería de mostrar los nombres de las columnas y numeración de las filas junto con una pequeña porción de los datos.

In [3]:
```
# Tiempo estimado de ejecución: 2 segundos aprox.

%%R
### Lectura
data <- read.csv(url("https://drive.google.com/uc?id=1GO2NBxYw54K6HkN-YgXbNadrLo
5O6-0u"))

### Visualización de una pequeña parte de los datos
head(data)
```

```
  Sex  Age Tanner Height   BMI   WC TAGmgDL HDLCmgDL LDLCmgDL SBP DBP Sedentary
1   1  9.5      0   1.55 11.34 60.0      55       51       93  97  60  411.0893
2   1  8.0      0   1.15 12.40 46.3      51       70       59  90  55  435.6071
3   0 10.5      0   1.42 12.99 67.5      65       60       96  96  54  483.9048
4   0  8.1      0   1.27 13.43 53.1      41       78      100 108  46  429.2976
5   1 10.4      0   1.32 13.72 51.9      39      100      120 107  69  512.0714
6   0 10.4      0   1.29 14.02 54.9      57       76       73  87  59  451.2321
     Light Moderate  Vigorous HOMA
1 321.5804 22.13393  3.982143 1.98
2 316.9762 48.05952 14.273810 0.87
3 337.7857 33.30952  7.988095 1.46
4 241.9762 39.67857 11.821429 1.07
5 216.0357  9.75000  2.410714 0.80
6 257.6429 36.40179  9.767857 1.35
```

REFERENCIAS BIBLIOGRÁFICAS

- Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani. An Introduction to Statistical Learning with Applications in R Springer, 2013 (**Chapter 03**)
- McDonald, J.H. Handbook of Biological Statistics (3rd ed.). Sparky House Publishing, Baltimore, Maryland, 2014. Pages 190-208 in the printed version
- Usando rpy2 en notebooks: https://rpy2.github.io/doc/latest/html/notebooks.html (https://rpy2.github.io/doc/latest/html/notebooks.html)
- Usando read.csv de R: https://www.rdocumentation.org/packages/utils/versions/3.6.2/topics/read.table (https://www.rdocumentation.org/packages/utils/versions/3.6.2/topics/read.table)

REFERENCIAS ADICIONALES

- M.J. Gacto, J.M. Soto-Hidalgo, J. Alcalá-Fdez, and R. Alcalá (2019). Experimental Study on 164 Algorithms Available in Software Tools for Solving Standard Non-Linear Regression Problems. IEEE Access 7, 2019, pp. 108916-108939; https://doi.org/10.1109/ACCESS.2019.2933261 (https://doi.org/10.1109/ACCESS.2019.2933261)