

Introducción al Aprendizaje Automático: Modelos Lineales

Alberto Torres Barrán

29 de Mayo del 2018

Índice

1. Introducción
2. Preproceso de datos
3. Modelos lineales básicos
 - Regresión lineal
 - Regresión logística
4. Extensiones
 - Modelos lineales generalizados (GLM)
 - Regularización
 - Modelos aditivos generalizados (GAM)
5. Análisis de resultados

Índice

1. Introducción
2. Preproceso de datos
3. Modelos lineales básicos
 - Regresión lineal
 - Regresión logística
4. Extensiones
 - Modelos lineales generalizados (GLM)
 - Regularización
 - Modelos aditivos generalizados (GAM)
5. Análisis de resultados

¿Qué es el *Machine Learning*?

De la Wikipedia:

*Machine learning is a subfield of **computer science** that evolved from the study of **pattern recognition** and computational learning theory in **artificial intelligence**. In 1959, Arthur Samuel defined machine learning as a “Field of study that gives **computers the ability to learn without being explicitly programmed**”. Machine learning explores the study and construction of algorithms that can learn from and make predictions on data.*

Está íntimamente ligado con otras disciplinas.

THIS IS YOUR MACHINE LEARNING SYSTEM?

YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.

WHAT IF THE ANSWERS ARE WRONG?

JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.



Fuente

Data Science **Data Mining**
Statistics
Artificial Intelligence
Machine Learning
Pattern Recognition

Statistics Más antigua (aprox. 1749), el resto de disciplinas utilizan algunas de sus técnicas: estadística descriptiva, análisis de regresión, inferencia.

Artificial Intelligence Más moderna, 1940. Algunos problemas que intenta resolver: procesamiento lenguaje natural, planificación, visión por computador, robótica.

Machine Learning Rama de la IA, 1946. Se utiliza para resolver algunos de los problemas que tiene la IA.

Pattern Recognition En general se usa como sinónimo de *Machine Learning*.

Data Mining Técnicas de modelado estadístico y *machine learning* aplicadas a un dominio en concreto.

Data Science Término más moderno, mezcla de todo lo anterior.

De la Wikipedia:

Data science employs techniques and theories drawn from many fields within the broad areas of mathematics, statistics, information science, and computer science, including signal processing, probability models, machine learning, statistical learning, data mining, database, data engineering, pattern recognition and learning, visualization, predictive analytics, uncertainty modeling, data warehousing, data compression, computer programming, artificial intelligence, and high performance computing.

La conclusión es que se tratan de un perfil muy amplio, con un conjunto de habilidades poco definido.

- ▶ En 2012, Harvard Business Review publicó el artículo:

DATA

Data Scientist: The Sexiest Job of the 21st Century

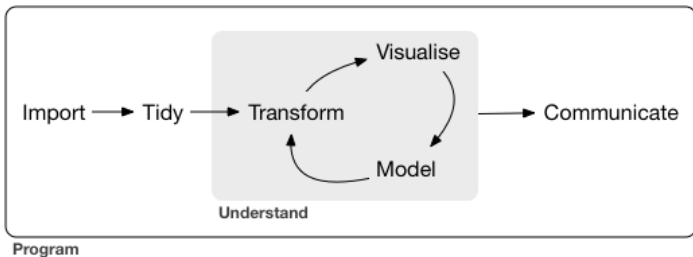
by Thomas H. Davenport and D.J. Patil

FROM THE OCTOBER 2012 ISSUE

- ▶ La consultora [McKinsey](#) calcula que para 2018 habrá una demanda de entre 140,000-190,000 puestos de *data science* sin cubrir.
- ▶ Nadie sabe muy bien lo que es pero todo el mundo quiere uno.

Flujo de trabajo de un equipo de *data science*

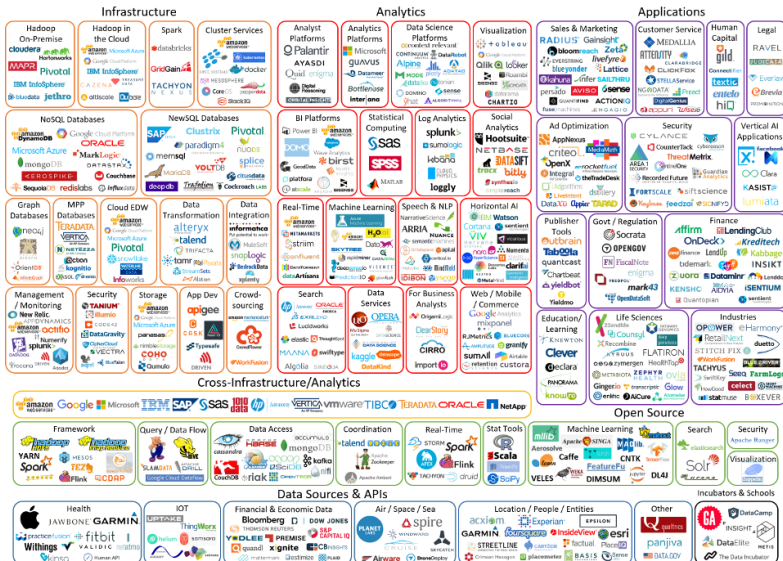
- ▶ En ocasiones un único perfil realiza todas las tareas.
- ▶ Sin embargo, cada vez es más habitual tener un equipo donde cada integrante esté especializado en distintas partes del proceso.



Fuente: Hadley Wickham, [R for Data Science](#)

Herramientas de análisis

Big Data Landscape 2016 (Version 2.0)

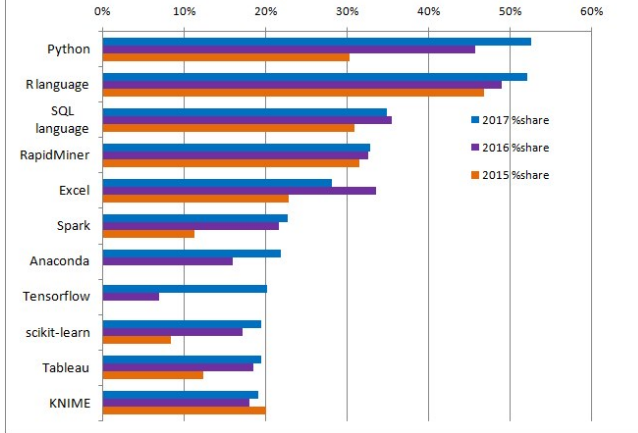


Last Updated 2/12/2016

© Matt Turck (@mattturck), Jim Hao (@jimrhao), & FirstMark Capital (@firstmarkcap)

FIRSTMARK

KDnuggets Analytics, Data Science, Machine Learning Software Poll, top tools share, 2015-2017



Software para proyectos de *analytics*, y *data science*. Encuesta de KDnuggets.

1. Jerome H. Friedman. *Data Mining and Statistics: What's the Connection?* (1998). [\[url\]](#)
2. Leo Breiman. *Statistical Modeling: The Two Cultures* (2001). [\[url\]](#)
3. Cross Validated. *What is the difference between data mining, statistics, machine learning and AI?* (2010). [\[url\]](#)
4. Sakthi Dasan Sekar. *What is the difference between Artificial Intelligence, Machine Learning, Statistics, and Data Mining* (2014). [\[url\]](#).
5. Cross Validated. *What exactly is Big Data?* (2015). [\[url\]](#)
6. David Donoho. *50 years of Data Science* (2015). [\[url\]](#)

Tipos de aprendizaje

- ▶ **Supervisado:** dados pares de entrada-salida, el objetivo es inferir su relación
Ejemplos: regresión lineal, regresión logística
- ▶ **No supervisado:** dados unos datos de entrada, el objetivo es inferir cierta estructura inherente en los mismos, sin necesidad de especificar las salidas de forma explícita
Ejemplos: clustering, reducción de dimensión
- ▶ Existen otros tipos de tareas en los que el acceso a las salidas está limitado de distintas formas:
 - ▶ Aprendizaje activo
 - ▶ Aprendizaje semi-supervisado
 - ▶ Aprendizaje por refuerzo

Aprendizaje supervisado

- ▶ Dado un conjunto de datos, compuesto por observaciones de diferentes variables, llamamos **variable respuesta** a aquella que es objeto del estudio.
- ▶ Una vez identificada la respuesta, tenemos dos objetivos principales:
 - ▶ Predicción: ser capaz de predecir cual va a ser la respuesta para observaciones futuras.
 - ▶ Información: extraer información sobre la relación de la variable de respuesta con el resto.
- ▶ A su vez distinguimos dos tipos de problemas:
 - ▶ Regresión: la variable respuesta es continua.
 - ▶ Clasificación: la variable respuesta es discreta (número finito de categorías).

Índice

1. Introducción
2. Preproceso de datos
3. Modelos lineales básicos
 - Regresión lineal
 - Regresión logística
4. Extensiones
 - Modelos lineales generalizados (GLM)
 - Regularización
 - Modelos aditivos generalizados (GAM)
5. Análisis de resultados

Primeros pasos

- ▶ Los datos a analizar a menudo provienen de fuentes variadas (redes sociales, sensores, encuestas, ...) y están almacenados en diferentes soportes (ficheros de texto, base de datos, ficheros binarios, *streams...*).
- ▶ Lo primero es identificar el problema qué queremos resolver y cuales son las variables que tenemos disponibles y pueden aportar información.
- ▶ Ante la duda, no descartar variables/información ni observaciones antes de tiempo.
- ▶ Lo segundo es combinar todas esa información y transformarla en una mezcla de variables numéricas (valores continuos) y categóricas (valores discretos).
- ▶ El objetivo final del preproceso es organizar esos datos en un formato tabular (filas y columnas).

Distintos tipos de información

- ▶ En ocasiones no es trivial transformar ciertos tipos de información en variables numéricas y/o categóricas.
- ▶ Para estos casos a menudo es necesario un preproceso extra, muy dependiente del problema a resolver y específico del dominio.
- ▶ Ejemplos típicos:
 - ▶ Texto (tweets, páginas web, documentos): *word2vec*, *bag-of-words*, modelos *n-gram*.
 - ▶ Imágenes: valores RGB de los píxeles, intensidad de gris.
 - ▶ Audio: transformada de Fourier, coeficientes MFCC.
 - ▶ Video: secuencia de *frames*.

- ▶ Es importante distinguir cuando una variable tiene valor 0 ó no conocido.
- ▶ Estos valores pueden venir representados por múltiples caracteres (“*”, “-”, campo vacío, etc.).
- ▶ Hay que codificarlos de manera especial para tenerlos en cuenta en los análisis.
- ▶ En general,
 1. Si tenemos suficientes datos, podemos simplemente ignorar las observaciones en las que falte alguno.
 2. Sino, podemos completar dichas observaciones que faltan con, por ejemplo, la mediana del resto.
- ▶ **Ejemplo:** en datos que provienen de un reconocimiento médico varios pacientes no tienen ningún valor en el campo de “Fármacos”. No toman ninguna medicación o el médico no ha registrado la respuesta?

- ▶ Distinguir si un valor es erróneo o válido pero extremo es muy complicado y dependiente del dominio.
- ▶ Existen test estadísticos que permiten identificarlos.
- ▶ En general no son perjudiciales, pero si es importante corregir los valores que son **imposibles**.
- ▶ **Ejemplo:** en datos provenientes de un reconocimiento médico, aparece un paciente con un IMC de 50.

Normalización

- ▶ Las variables numéricas suelen tener rangos muy diversos.
- ▶ Ejemplo: salario (1000 – 1000000 €) y edad (0–100).
- ▶ Algunos modelos interpretan esta diferencia de escalas como que unas variables son más importantes que otras.
- ▶ Existen varias normalizaciones para que estas variables sean comparables:
 - ▶ Media 0 varianza 1.
 - ▶ Escalar al intervalo $-1, 1$.
 - ▶ ...
- ▶ En ocasiones normalizar las variables también puede ayudar a que el proceso de aprendizaje sea más rápido.
- ▶ Cuidado al analizar los resultados, ya que están en los nuevos rangos.

Variables categóricas

- ▶ Muy comunes en distintas fuentes de datos.
- ▶ Importante tenerlas en cuenta para la fase de preproceso y análisis previo.
- ▶ Muy pocos algoritmos son capaces de tratarlas directamente.
- ▶ Por tanto, antes de pasar a la siguiente etapa (modelado) queremos convertirlas en numéricas.
- ▶ La transformación donde se asigna a cada uno de sus valores un número entero no suele ser buena idea, ya que crea una relación artificial de orden y falsea las distancias.
- ▶ Una forma es utilizar una codificación “*dummy*”.

Ejemplo codificación “*dummy*”

Edad	Sexo		Edad	Es mujer?	Es hombre?
34	H	\Rightarrow	34	0	1
18	M		18	1	0
67	M		67	1	0
21	M		21	1	0
15	H		15	0	1

- ▶ Finalmente, podemos eliminar una de las dos nuevas variables puesto que están perfectamente correladas.
- ▶ En general, para una variable categórica con p valores añadimos $p - 1$ variables nuevas.

Otras codificaciones

- ▶ Si en la semántica de la variable hay implícita una relación de orden: Puntuación {baja, media, alta} \Rightarrow {1,2,3}.
- ▶ Si hay relación de orden y no queremos falsear las distancias:

Mes	Día	Temp.		Días desde 01/01	Temp.
Enero	29	22.2	\Rightarrow	29	22.2
Enero	30	27.8		30	27.8
Enero	31	28.6		31	28.6
Febrero	1	26.1		32	26.1
Febrero	2	25.3		33	25.3

Factores

- ▶ Un factor es un tipo de dato que se utiliza para codificar valores categóricos, por ejemplo: `renta = {alta, baja, media}`.
- ▶ Se crean con la función `factor`:

```
> f <- factor(c("hombre", "mujer", "mujer"))
```
- ▶ Se pueden ver los niveles (valores distintos) con la función `levels()`:

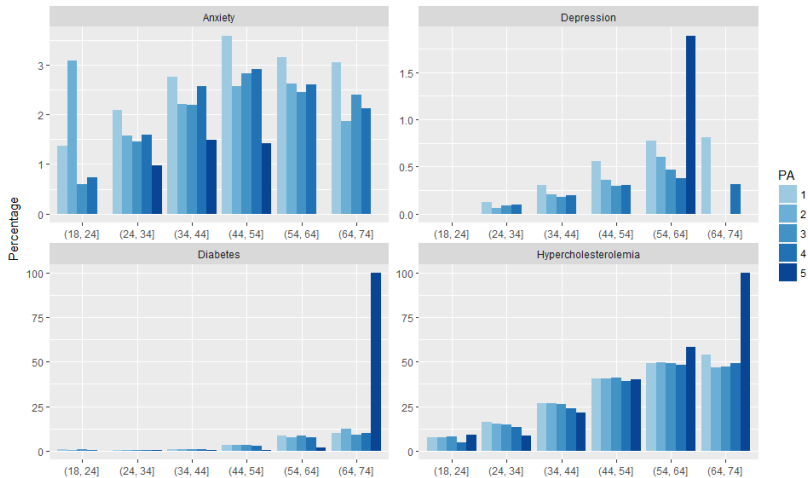
```
> levels(f)
[1] "hombre" "mujer"
```
- ▶ Función `relevel()`: reordena los niveles del factor especificado, poniendo el nivel especificado de primero

Operaciones con factores

- ▶ Función `cut()`: crea un factor dividiendo en rangos un vector numérico de acuerdo a unos puntos de corte
- ▶ Función `tapply()`: aplica una función a cada uno de los elementos de un vector, divididos en los distintos grupos de un determinado factor.
- ▶ Función `by()`: similar a la anterior, pero el objeto sobre el que se aplica la operación agrupada puede ser un `data.frame`.
- ▶ Función `aggregate()`: similar a `tapply()` pero devuelve un `data.frame` y acepta “fórmulas”.
- ▶ Funciones `table()`, `prop.table()`, `margin.table()` y `xtabs()`: crear tablas de contingencia a partir de ciertos factores

- ▶ En ocasiones es útil hacer gráficos de algunas variables para ver que tipo de relación tienen con la respuesta.
- ▶ Sin embargo, a medida que los conjuntos de datos son más grandes:
 1. el número de variables puede ser muy grande, incluso del orden de millones.
 2. la relación de las variables de entrada es muy compleja y altamente no lineal.
- ▶ En esos casos es muy difícil hacer gráficos que proporcionen información relevante, ya que podemos representar como mucho 2 o 3 dimensiones.
- ▶ Los gráficos múltiples (facetas) y algunas transformaciones estadísticas (reducción de dimensionalidad) pueden aliviar el problema.

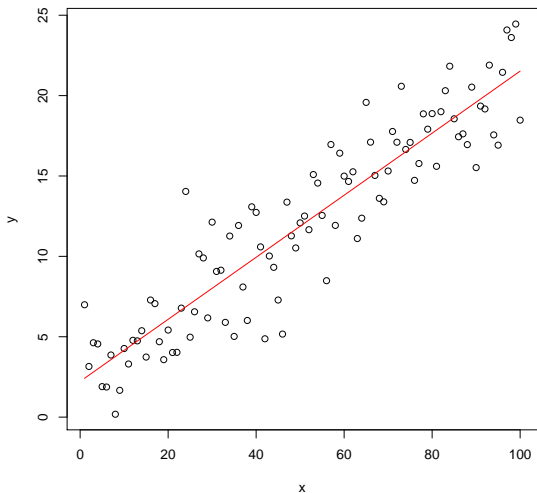
Ejemplo



Índice

1. Introducción
2. Preproceso de datos
3. Modelos lineales básicos
 - Regresión lineal
 - Regresión logística
4. Extensiones
 - Modelos lineales generalizados (GLM)
 - Regularización
 - Modelos aditivos generalizados (GAM)
5. Análisis de resultados

Regresión lineal: ejemplo



Regresión lineal

- ▶ La regresión lineal asume que la variable de respuesta y depende linealmente de las variables independientes,

$$y = w_0 + x_1w_1 + x_2w_2 + x_3w_3 + \cdots + x_dw_d$$

- ▶ Si tenemos n observaciones de cada una de las variables, podemos escribirlo en notación matricial

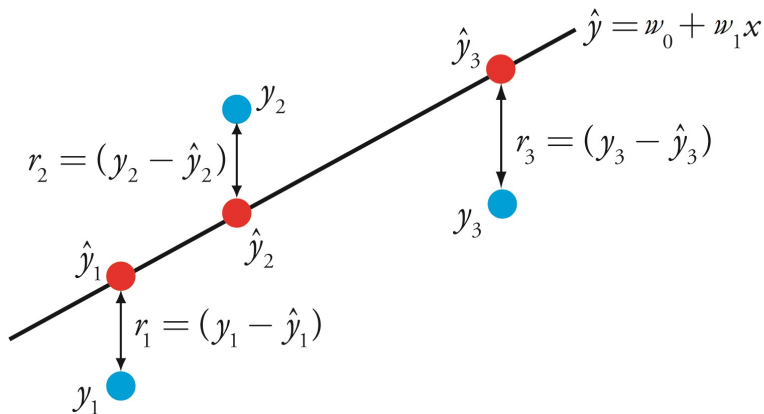
$$\mathbf{y} = \mathbf{X}\mathbf{w}$$

- ▶ El término de bias w_0 se incluye como una columna constante de 1s en \mathbf{X}
- ▶ El objetivo es estimar los pesos o coeficientes w

Hipótesis de la regresión lineal

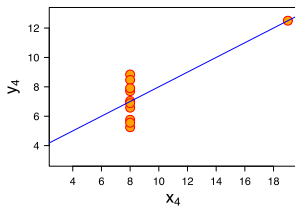
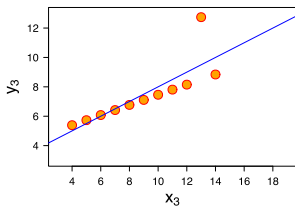
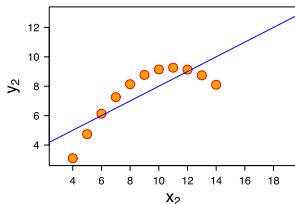
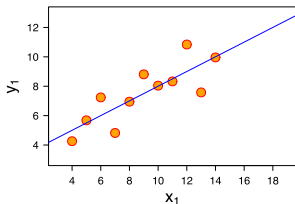
- ▶ El método más común para calcular la recta de regresión se conoce como mínimos cuadrados.
- ▶ Teóricamente, para que el ajuste esté bien definido se asume que:
 1. la respuesta depende linealmente de las variables
 2. el modelo está especificado correctamente (no faltan variables)
 3. hay menos variables que observaciones
 4. no hay dos variables con correlación perfecta
- ▶ Es un modelo predictivo, ya que nos permite calcular el valor de y para nuevos valores de x .

Mínimos cuadrados



Cuarteto de Anscombe

La pregunta es, ¿cómo de bien se ajusta la recta a nuestros datos?



Bondad de ajuste

- ▶ Históricamente se medía la calidad del modelo ó “bondad del ajuste” con diversos test sobre los residuos:
 - ▶ Homocedásticos (varianza constante)
 - ▶ Media cero
 - ▶ Sin autocorrelación
 - ▶ (Distribución normal)
- ▶ El modelo puede ser útil a pesar de que las hipótesis de la regresión y los test sobre los residuos no se cumplan.
- ▶ A menudo nos interesa únicamente la **capacidad predictiva**.
- ▶ “*All models are wrong, but some are useful*” (George Box).

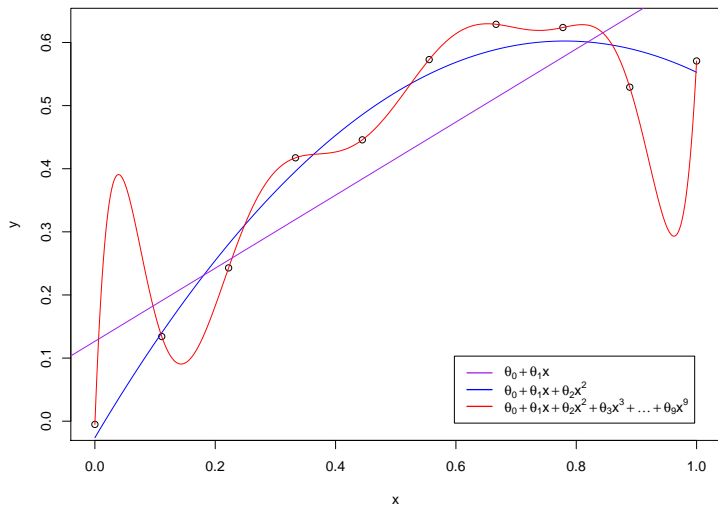
Minimización del riesgo empírico

- ▶ Necesitamos definir formalmente a que nos referimos con **capacidad predictiva**.
- ▶ Dadas unas variables \mathbf{x} , queremos encontrar una función $f(\mathbf{x})$ que se parezca lo máximo posible a la respuesta y .
- ▶ Para ello, necesitamos una función de pérdida $L(f(\mathbf{x}), y)$ que cuantifique cuando de diferente es nuestra predicción del valor real.
- ▶ El problema de aprendizaje consiste por tanto en encontrar la función f que minimiza la pérdida media de todas las observaciones:

$$\hat{f} = \arg \min \frac{1}{n} \sum_{i=1}^n L(f(\mathbf{x}_i), y_i)$$

- ▶ Si resolvemos el problema anterior con el error cuadrático como función de pérdida $L(f(\mathbf{x}), y) = (f(\mathbf{x}) - y)^2$, obtenemos el estimador de mínimos cuadrados.
- ▶ Por tanto, ya tenemos una forma de calcular el error de predicción de nuestro modelo original.
- ▶ La pregunta ahora es, podemos mejorar el modelo (disminuir su error)?
- ▶ Sí, aunque el modelo tiene que ser lineal en las variables, se pueden añadir nuevas variables que sean transformaciones polinómicas.
- ▶ De hecho, siempre podemos añadir expansiones polinómicas de forma que el error sea casi 0.

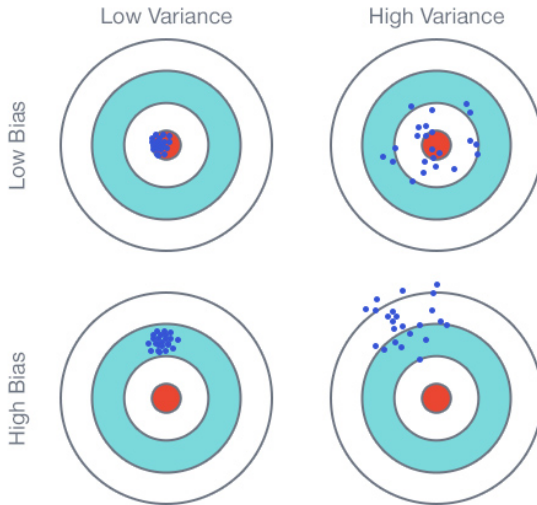
Sobreaajuste: ejemplo



Equilibrio sesgo-varianza: intuición

- ▶ Si el modelo es muy simple, la solución está sesgada y no ajusta bien los datos.
- ▶ Si el modelo es muy complejo, es muy sensible a pequeños cambios en los datos.
- ▶ En general el error calculado sobre las muestras usadas para entrenar el modelo (error de **entrenamiento**) es demasiado optimista.
- ▶ El error de entrenamiento se puede hacer arbitrariamente pequeño aumentando la complejidad del modelo.
- ▶ Nos interesa el error de **generalización**, es decir el error sobre muestras que el modelo no conoce.

Equilibrio sesgo-variance: definición gráfica



Fuente

Equilibrio sesgo-varianza: formulación

El error teórico de predicción es

$$\text{EP} = \mathbb{E}[(y - \hat{f}(x))^2].$$

Se puede descomponer en

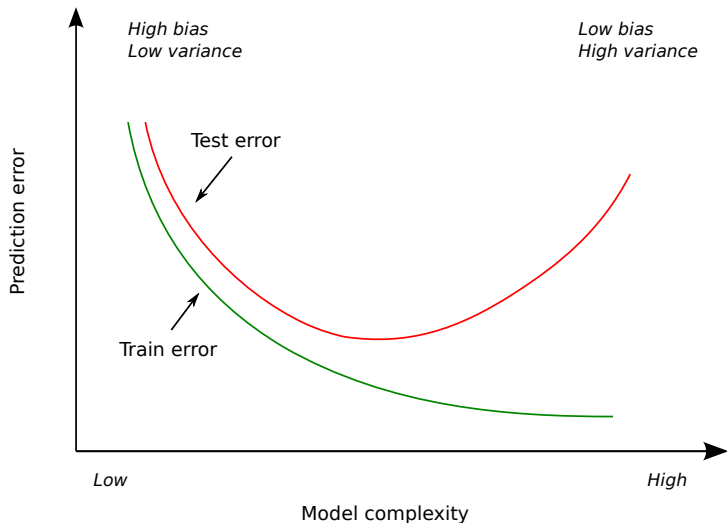
$$\text{EP} = \underbrace{\left(\mathbb{E}[\hat{f}(x)] - f(x)\right)^2}_{\text{Sesgo}^2} + \underbrace{E\left[\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\right]^2}_{\text{Varianza}} + \underbrace{\sigma^2}_{\text{Ruido}}$$

- ▶ Los términos de sesgo y varianza son opuestos: si disminuimos uno el otro aumenta y viceversa.
- ▶ El término del ruido es inherente a los datos y no podemos hacer nada.

Conjuntos de entrenamiento y test

- ▶ En la práctica, lo primero que hacemos cuando cargamos unos datos es dividirlos aleatoriamente en dos subconjuntos, entrenamiento y test.
- ▶ El subconjunto de test nos lo guardamos y no se utiliza **nunca** en la fase de aprendizaje del modelo.
- ▶ Una vez construido el modelo, se comprueba su rendimiento en el conjunto de test.
- ▶ Este último error es una buena estimación no sesgada de como se va a comportar nuestro modelo con nuevos datos.
- ▶ Existe una gran probabilidad de sobreajuste si el error de test es muy alto en comparación con el error de entrenamiento.

Error de predicción en función de la complejidad



Errores de regresión

Dado el valor real de la observación i , y_i y la predicción del modelo \hat{y}_i , podemos calcular:

- ▶ MAE (Mean absolute error)

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

- ▶ MSE (Mean squared error)

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

- ▶ En R un modelo lineal de la variable de respuesta y sobre las variables x_1, \dots, x_d , se define con la fórmula

$$y \sim x_1 + x_2 + \dots + x_d$$

- ▶ Para ajustar un modelo lineal por mínimos cuadrados se usa la función `lm()`, pasando como primer argumento la fórmula anterior.
- ▶ Las fórmulas también pueden contener expresiones aritméticas de variables (expansiones polinómicas, logaritmos, etc).
- ▶ Ejemplo: modelo de regresión de la variable `mpg` sobre `wt`

```
> fit <- lm(mpg ~ wt, data=mtcars)
```

Fórmulas

- ▶ Las fórmulas son objetos especiales de R que representan relaciones simbólicas entre variables:
`respuesta ~ variables independientes`
- ▶ Se usan en funciones como `aggregate()`, `boxplot()`, y `lm()`.
- ▶ Los operadores aritméticos tienen otro significado cuando se usan dentro de las fórmulas. Ejemplos:

$$y \sim u + v + w + u:v + u:w + v:w$$

$$y \sim u * v * w - u:v:w$$

$$y \sim (u + v + w)^2$$

- ▶ Si queremos que tengan su significado habitual tenemos que utilizar el operador `I()`:

$$y \sim u + v + w + I(u*v) + I(u*w) + I(v*w)$$

Ejercicio regresión

Con el conjunto de datos **diamonds**:

- ▶ Separarlos aleatoriamente en un 60% de entrenamiento y un 40% de test.
- ▶ Ajustar un modelo lineal del precio sobre los quilates.
- ▶ Ajustar un modelo cuadrático con las mismas variables.
- ▶ Calcular el error cuadrático medio sobre el conjunto de entrenamiento y test, definido como

$$\text{ECM} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2,$$

donde y es la variable respuesta y \hat{y} la predicción del modelo.

- ▶ ¿Qué pasa ahora si ajustamos el modelo sobre todas las variables? ¿Disminuye el error?

Regresión logística

- ▶ Es un modelo lineal para problemas de clasificación.
- ▶ En lugar de una variable continua, la respuesta es ahora una variable discreta con 2 o más valores, que se denominan *clases*.
- ▶ En el caso binario, el modelo estima la probabilidad de que cada uno de los ejemplos pertenezca a la clase 0 o 1.
- ▶ Finalmente se predice la clase 0 si la probabilidad es menor que 0.5 y la clase 1 en caso contrario.
- ▶ Se puede ver como una caso especial del modelo lineal generalizado.

Regresión logística: formulación

- ▶ La fórmula de la regresión logística es

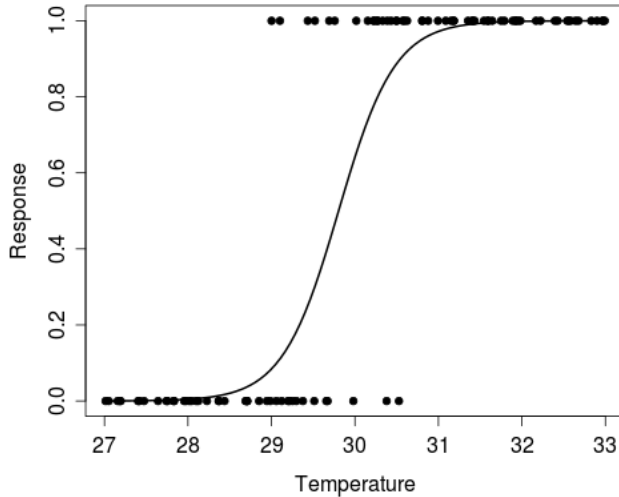
$$y = \sigma(w_0 + w_1x_1 + w_2x_2 + \dots + w_dx_d)$$

donde $\sigma(\cdot)$ es la función logística

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

- ▶ La función logística siempre tiene como salida un número en el intervalo $(0, 1)$
- ▶ Por tanto, interpretamos la salida del modelo como la probabilidad de pertenecer a una clase o a la otra (clasificación binaria)

Regresión logística: ejemplo



Interpretación de coeficientes: odds ratio

- ▶ Dado un modelo lineal con una única variable independiente, el odds ratio se define como

$$\text{OR} = \frac{\exp(w_0 + w_1(x + 1))}{\exp(w_0 + w_1x)} = \exp(w_1)$$

- ▶ Es decir, como cambian las probabilidades de la salida cuando la variable x aumenta una unidad:
 - ▶ Si $\text{OR} = 1$ la variable x no tiene ninguna asociación con la salida
 - ▶ Si $\text{OR} > 1$ la variable está asociada con una mayor probabilidad de la salida
 - ▶ Si $\text{OR} < 1$ la variable está asociada con una menor probabilidad de la salida

Errores de clasificación

Los principales errores de clasificación se pueden calcular a partir de la **matriz de confusión**:

		Predicted class	
		<i>P</i>	<i>N</i>
Actual Class	<i>P</i>	True Positives (TP)	False Negatives (FN)
	<i>N</i>	False Positives (FP)	True Negatives (TN)

$$\text{Accuracy: } \frac{TP+TN}{P+N}$$

$$\text{Sensitivity, recall, TPR: } \frac{TP}{TP+FN}$$

$$\text{Specificity, TNR: } \frac{TN}{TN+FP}$$

$$\text{Precision, PPV: } \frac{TP}{TP+FP}$$

$$\text{F1 score: } 2 \times \frac{PPV \times TPR}{PPV + TPR}$$

Índice

1. Introducción
2. Preproceso de datos
3. Modelos lineales básicos
 - Regresión lineal
 - Regresión logística
4. Extensiones
 - Modelos lineales generalizados (GLM)
 - Regularización
 - Modelos aditivos generalizados (GAM)
5. Análisis de resultados

Modelos lineales generalizados (GLMs)

- Generalización de la regresión lineal que permite distribuciones de errores distintas de la distribución normal.
- Se asume que la media de dicha distribución depende de las variables independientes de la siguiente forma:

$$\mathbb{E}(\mathbf{y}) = \mu = g^{-1}(\mathbf{X}\mathbf{w})$$

donde $\mathbb{E}(\cdot)$ es el valor esperado y g es la función de enlace

- La función de enlace proporciona la relación entre la media de la distribución y el predictor lineal

Ejemplo: distribución de Bernoulli

- ▶ Cuando la distribución de la salida \mathbf{y} es una Bernoulli el modelo se conoce con el nombre de regresión logística
- ▶ La función de media es la logística,

$$\mu = g^{-1}(\mathbf{X}\mathbf{w}) = \frac{1}{1 + \exp(-\mathbf{X}\mathbf{w})}$$

- ▶ La función de enlace es la inversa de la anterior,

$$\mathbf{X}\mathbf{w} = g(\mu) = \ln \left(\frac{\mu}{1 - \mu} \right)$$

- ▶ Para cada distribución, hay una función de enlace “canónica” que es la que se usa habitualmente

Ejemplo: distribución de Poisson

- ▶ Esta distribución está indicada cuando queremos modelizar una variable de salida entera y no real (por ej. conteos)
- ▶ Función de media

$$\mu = \exp(\mathbf{X}\mathbf{w})$$

- ▶ Función de enlace

$$\mathbf{X}\mathbf{w} = \ln(\mu)$$

- ▶ Otras distribuciones posibles son la Gamma, Exponencial, Multinomial, etc.

- ▶ La función para ajustar modelos lineales generalizados es `glm()`
- ▶ Tiene los mismos argumentos principales que `lm()`, pero además tenemos que especificar la distribución de la variables dependiente con el parámetro `family`
- ▶ Ejemplo: regresión logística

```
> fit <- glm(Species ~ Petal.Length, data=iris,
              family=binomial)
```
- ▶ Por defecto se usa la función de enlace “canónica”, pero esto se puede modificar (ver ayuda)

- ▶ El estimador de mínimos cuadrados para la regresión lineal es el *mejor* estimador no sesgado, donde *mejor* se refiere al que tiene menor varianza.
- ▶ Sin embargo, a menudo se puede reducir esta varianza bastante, en detrimento de introducir un pequeño sesgo.
- ▶ Esto se consigue limitando la complejidad del modelo con un término de **regularización**.
- ▶ Un ejemplo muy común es Ridge Regression, que añade una regularización l_2 a mínimos cuadrados:

$$\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{w}\|_2^2$$

- ▶ Similar a Ridge Regression, Lasso añade un término de regularización l_1 :

$$\min_{\mathbf{w}} ||\mathbf{X}\mathbf{w} - \mathbf{y}||_2^2 + \lambda ||\mathbf{w}||_1$$

donde

$$||\mathbf{w}||_1 = \sum_{i=1}^d |w_i|$$

- ▶ El valor absoluto promueve que muchos coeficientes sean 0 después de ajustar el modelo
- ▶ Dichos coeficientes no tienen por tanto efecto en la salida
- ▶ Se podría considerar que son variables “poco importantes”

- Combina las regularizaciones de Ridge y Lasso:

$$\min_{\mathbf{w}} ||\mathbf{X}\mathbf{w} - \mathbf{y}||_2^2 + \lambda_1 ||\mathbf{w}||_1 + \lambda_2 ||\mathbf{w}||_2^2$$

- Otra forma de escribirlo:

$$\min_{\mathbf{w}} ||\mathbf{X}\mathbf{w} - \mathbf{y}||_2^2 + \lambda(\alpha ||\mathbf{w}||_1 + (1 - \alpha) ||\mathbf{w}||_2^2)$$

- Para $\alpha = 0$ recuperamos Ridge Regression y para $\alpha = 1$ el Lasso
- Mantiene la dispersión en los coeficientes del Lasso pero en general se obtienen mejores modelos en términos de error
- Problema: tenemos que decidir el valor de dos hiper-parámetros (α y λ)

Paquete `glmnet` y `glmnetUtils`

- ▶ Implementa GLMs con regularización Lasso y Elastic Net
- ▶ Muy eficiente (escrito en Fortran)
- ▶ Se pueden ajustar no solo regresiones lineales con regularización, sino también regresiones logísticas, regresiones de Poisson, etc.
- ▶ Implementa también un mecanismo para seleccionar automáticamente el parámetro λ (pero no α) usando validación cruzada
- ▶ `glmnet` no tiene interfaz para fórmulas, pero `glmnetUtils` incorpora una

Ejercicio titanic I

Vamos a intentar predecir la supervivencia de las víctimas del Titanic a partir de las siguientes variables:

- ▶ **survival**: Supervivencia (0 = No; 1 = Si)
- ▶ **pclass**: Clase de pasajero (1, 2, 3)
- ▶ **name**: Nombre
- ▶ **sex**: Sexo
- ▶ **age**: Edad
- ▶ **sibsp**: Número de hermanos/esposos/as a bordo.
- ▶ **parch**: Número de padres/hijos a bordo
- ▶ **ticket**: Número de ticket
- ▶ **fare**: Coste del billete
- ▶ **cabin**: Cabina
- ▶ **embarked**: Puerto de embarque

Ejercicio titanic II

1. Cargar el fichero `titanic.csv` en R.
2. Ver cuantos valores *missing* tiene cada variable con `summary`.
3. Eliminar la variable `Cabin` (¿por qué?).
4. Eliminar también las variables `PassengerId`, `Name` y `Ticket` (¿por qué?).
5. Eliminar las filas que contengan algún NA (función `na.omit`).
6. Convertir la variable `Survived` a un factor.
7. Dividir datos en 80% `entrenamiento` y 20% `test`, aleatoriamente.
8. Ajustar un modelo de regresión logística estándar y otro con regularización (función `glm` y paquete `glmnet`).

Modelos aditivos generalizados (GAM)

- ▶ Extensión de los GLMs donde la salida depende linealmente de funciones de las variables predictoras:

$$\mathbb{E}(y) = g^{-1}(f_1(x_1) + f_2(x_2) + f_3(x_3) + \dots + f_d(x_d))$$

- ▶ Las funciones pueden ser distintas para cada una de las variables
- ▶ Pueden ser paramétricas o funciones generales “suaves”
- ▶ La mayoría de implementaciones modernas restringen las funciones para que sean de la forma

$$f_i(x_i) = \sum_{k=1}^{K_i} \beta_{ik} b_{ik}(x_i)$$

donde $b_{ik}(x_i)$ son funciones de base

- ▶ Implementación de GAMs en R, alternativa al clásico paquete `gam`
- ▶ Varias funciones de suavizado posible
- ▶ Incluye el grado de “suavizado” en el ajuste del modelo (no es necesario especificarlo)
- ▶ Incluye múltiples distribuciones (Binomial, Bernoulli, Poisson, etc.)
- ▶ Ejemplo:

```
> fit <- gam(mpg ~ s(wt), data=mtcars)
```

Ejercicio

Con los datos de diamantes del ejercicio anterior:

- ▶ Ajustar un GAM del precio sobre los quilates
- ▶ Comparar el error sobre el conjunto de test con el modelo lineal y con el modelo lineal añadiendo una nueva variable que sea el cuadrado del precio
- ▶ Ajustar un modelo Lasso modelizando el precio usando el resto de variables de entrada
- ▶ ¿Cuál es el valor del λ óptimo?
- ▶ ¿Qué variables son las más importantes y cuales son innecesarias?

Índice

1. Introducción
2. Preproceso de datos
3. Modelos lineales básicos
 - Regresión lineal
 - Regresión logística
4. Extensiones
 - Modelos lineales generalizados (GLM)
 - Regularización
 - Modelos aditivos generalizados (GAM)
5. Análisis de resultados

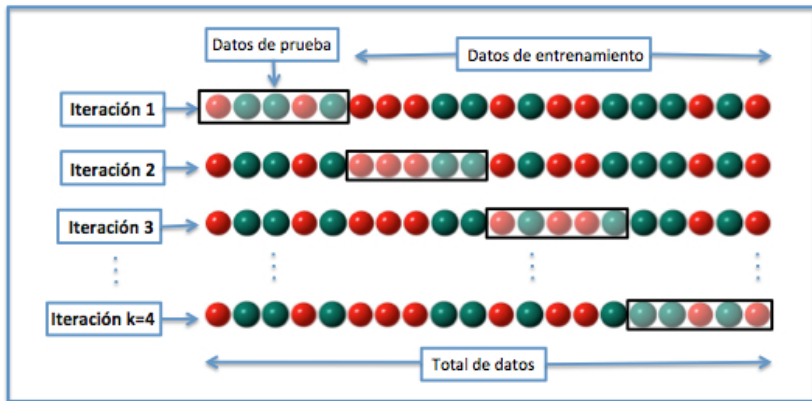
- ▶ Si el resultado del modelo no es tan bueno como nos gustaría no hay que perder la esperanza, ya es muy común que el análisis se realice de forma iterativa.
- ▶ Podemos probar varias cosas:
 1. Ajustar mejor el modelo: muchos de los modelos que hemos visto tienen parámetros que influyen mucho en el rendimiento (más a continuación).
 2. Probar otros modelos: una opción muy común al hacer el análisis de unos datos es comenzar con modelos más simples e ir moviéndonos hacia modelos más complejos.
 3. Obtener más datos: no siempre es posible, pero en general con más datos se consigue mejor resultado que un algoritmo más inteligente.

Conjunto de validación

- ▶ Para escoger los valores de los parámetros de un modelo, podemos mirar el error de entrenamiento para varios valores y escoger el de menor error.
- ▶ Al hacer esto, hemos visto que el error de entrenamiento es una medida sesgada y por tanto los valores de los parámetros pueden no funcionar bien en el conjunto de test.
- ▶ Por tanto, necesitamos un tercer conjunto llamado conjunto de **validación** donde se va a medir el error para cada valor de los parámetros.
- ▶ Finalmente, escogemos el que tenga menor error de validación.
- ▶ El conjunto de test sigue sin tocar, y se usa como antes para estimar el error de generalización.

Validación cruzada

- Si no tenemos muchos datos y no queremos hacer un conjunto de validación, podemos usar validación cruzada.



Fuente

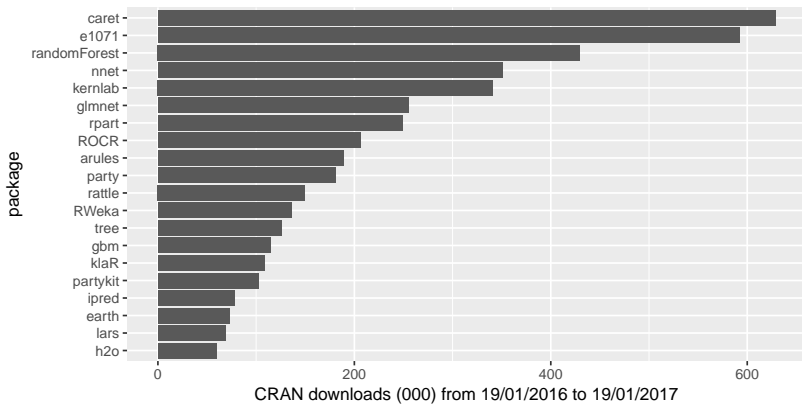
Selección de variables y reducción de dimensión

- ▶ Una forma de reducir el tiempo de computación de los algoritmos de aprendizaje es reduciendo el número de variables.
- ▶ Esto se puede hacer de dos maneras:
 1. Seleccionando únicamente un subconjunto de las variables de acuerdo a algún criterio de relevancia para predecir la respuesta.
 2. Proyectando nuestros datos a un espacio de menor dimensión.
- ▶ La primera opción tiene la ventaja de que las variables siguen siendo interpretables, es decir, tienen significado semántico.
- ▶ El método más común de proyección es PCA o análisis de componentes principales.

Colecciones de paquetes útiles

- ▶ En [este](#) enlace se puede ver una lista muy reciente de paquetes útiles.
- ▶ [Machine Learning in R](#) es una colección de paquetes de aprendizaje automático.
- ▶ [High-Performance computing in R](#) es una colección de paquetes de útiles para la computación de alto rendimiento.
- ▶ Recientemente aplicaciones web que permiten la visualización de resultados también gozan de gran popularidad, por ejemplo los notebooks de [Jupyter](#).
- ▶ En R destaca [Shiny](#), que permite convertir código R en aplicaciones web interactivas.

Paquetes más populares de ML



Descargas Enero 2016 - Enero 2017. [Fuente](#)

Resumen paquetes destacados

- ▶ Modelos lineales y utilidades:
 - ▶ `caret`, utilidades para clasificación y regresión
 - ▶ `MASS`, ridge regression
 - ▶ `ridge`, ridge regression con selección automática del hiper-parámetro
 - ▶ `glmnet`, GLMs con regularización Lasso o Elastic Net
 - ▶ `glmnetUtils`, utilidades para `glmnet`
 - ▶ `gam`, modelos aditivos generalizados
 - ▶ `mgcv`, modelos aditivos generalizados (recomendado)
- ▶ Algunos modelos más complejos:
 - ▶ `nnet`, Redes neuronales, también regresión logística multinomial
 - ▶ `e1071`, Support Vector Machines
 - ▶ `gbm`, Gradient Boosting
 - ▶ `randomForest`, Random Forest
 - ▶ `xgboost`, Extreme Gradient Boosting

- ▶ Conjunto de paquetes creados por Hadley Wickham que comparten una misma API y contienen funciones para el análisis de datos:
 - ▶ `ggplot2`, para hacer gráficos avanzados.
 - ▶ `dplyr`, para manipular datos.
 - ▶ `tidyr`, para limpiar datos.
 - ▶ `readr`, para importar datos.
 - ▶ `purrr`, para programación funcional.
 - ▶ `tibble`, implementa *tibbles*, una versión moderna de los `data.frames`.
- ▶ El paquete *tidyverse* instala y carga los paquetes anteriores.
- ▶ También instala otros paquetes que pueden ser útiles aunque no los carga por defecto.
- ▶ Para más información y la lista completa de paquetes:
<https://github.com/tidyverse/tidyverse>.

Libros y manuales

En general, se pueden encontrar muchos manuales en las secciones *Manuals* y *Contributed* de [CRAN](#), así como ejemplos en la web [R Pubs](#). Algunos recursos más específicos:

- Libros**
- ▶ R for Data Science [\[url\]](#).
 - ▶ An Introduction to Statistical Learning with Applications in R [\[url\]](#).

- E-Books**
- ▶ YaRrr! The Pirate's Guide to R [\[url\]](#).
 - ▶ The R Inferno [\[url\]](#).
 - ▶ R Programming [\[url\]](#).

- Blogs**
- ▶ RTutorial [\[url\]](#).
 - ▶ Quick-R [\[url\]](#).
 - ▶ RStudio [\[url\]](#).
 - ▶ RBloggers [\[url\]](#).

FAQs y comunidades

- ▶ [StackOverflow](#): las preguntas con el tag R contienen mucha información y problemas resueltos. Además, las nuevas preguntas se responden en cuestión de horas.
- ▶ [CrossValidated](#): no es una comunidad específica de R (más bien de estadística), pero hay mucha información acerca de cómo realizar procedimientos concretos de análisis de datos y aprendizaje automático en R.
- ▶ [@RLangTip](#): Twitter que publica consejos y trucos diarios.
- ▶ [R Programming for Data Analysis](#): Comunidad de Google+.
- ▶ [Statistics and R](#): Otra comunidad de Google+.
- ▶ [The R Project for Statistical Computing](#): Grupo de LinkedIn.