Alumni

Lenguajes de programación

alumni.uoc.edu

¿Qué lenguajes de programación necesita el científico de datos?

¿Cuáles son los lenguajes utilizados hoy?

¿Cuáles son los lenguajes utilizados hoy?

El índice TIOBE, utilizado para medir la popularidad de los lenguajes, clasifica 250 lenguajes.

https://www.tiobe.com

Tipos de lenguajes

Compilados

- El **código fuente se transforma** a un ejecutable entendible para la máquina en determinada arquitectura **(HW+SO).**
- Las variables tienen que pertenecer a un tipo determinado (fuertemente tipadas).

Interpretados

- Utilizan funciones de más **alto nivel.**
- Son independientes de la plataforma.
- Los tipos de las variables **pueden ir cambiando** en forma dinámica durante la ejecución del programa.

Java (Maquina Virtual)

- Se consideran “compilados” porque **el fuente se transforma** para ser ejecutado por la máquina virtual en una arquitectura determinada.

Seleccionando los más utilizados en la **ciencias de datos** y considerando su popularidad...

LENGUAJE	Licencia	Tipo	Modelo	Extensiones	Conexiones	Otros
Python	PSFL	Interpretado Tipado dinámico	Objeto-Funcional-Imperativo (Multiparadigma)	C/C++	Librerías para conectar con BD SQL. Y para las aplicaciones Big Data más populares. Servicios web.	Bibliotecas de uso general. Fácil aprender
R	GPL	Interpretado.	Objeto-Funcional	R/C/C++ /Fortran	Librerías para conectar con BD SQL. Y para las aplicaciones Big Data más populares. Servicios web.	Comunidad estadística
C/C++	GPL	Compilado Fuertemente tipado	Imperativo /Multiparadigma	--	Librerías para conectar con BD SQL. No siempre hay interfaz con C, si con C++, Servicios web.	Eficiencia. Disponibilidad
Java/ JavaScript	GPL	Compilado / Interpretado	Objeto/ Multiparadigma	C / C++	Librerías para conectar con BD SQL. Y para las aplicaciones Big data más populares. Servicios web.	Portabilidad
GO	BSD	Compilado Fuertemente tipado	Objeto/Multipara-digma/concurrente	C	Librerías para conectar con BD SQL. Se están desarrollando librerías y conexiones. Servicios web.	Muy buen rendimiento y con facilidad de aprendizaje

Lenguajes recomendados para la ciencia de datos

Python

- Preferido por **ingenieros** y científicos en computación.
- Trabaja con todo tipo de datos.
- Lenguaje de **propósito general.**

https://pypi.python.org/pypi **128.324** paquetes

R

- Preferido por los **estadistas y matemáticos**
- Trabaja con datos filtrados y específicos
- Específico para **ciencia de datos.**

https://cran.r-project.org/ **12.088** paquetes

¿Dónde están los datos?

Orígenes de los datos

WEB: HTML/ XML

- Web *scrapping*
- Lenguajes con gestión de HTML / XML

Service Rest/ SOAP

- Web Services
- Manejo de JSON / XML-SOAP

Base de datos Relacionales

- SQL
- Lenguaje con conexión a DB

Repositorios NOSQL

- Mongo DB / ELKL / Dinamo DB /
- Logs, txt, CSV
- Amazon S3

Tipos de almacenamiento de datos

NOSQL

- Datos no estructurados
- Volumen extremadamente grande

Bases de datos Relacionales

- Datos estructurados
- Gran volumen de datos
- Cambio en los análisis a través del tiempo
- Publicar los análisis

Ficheros txt/csv

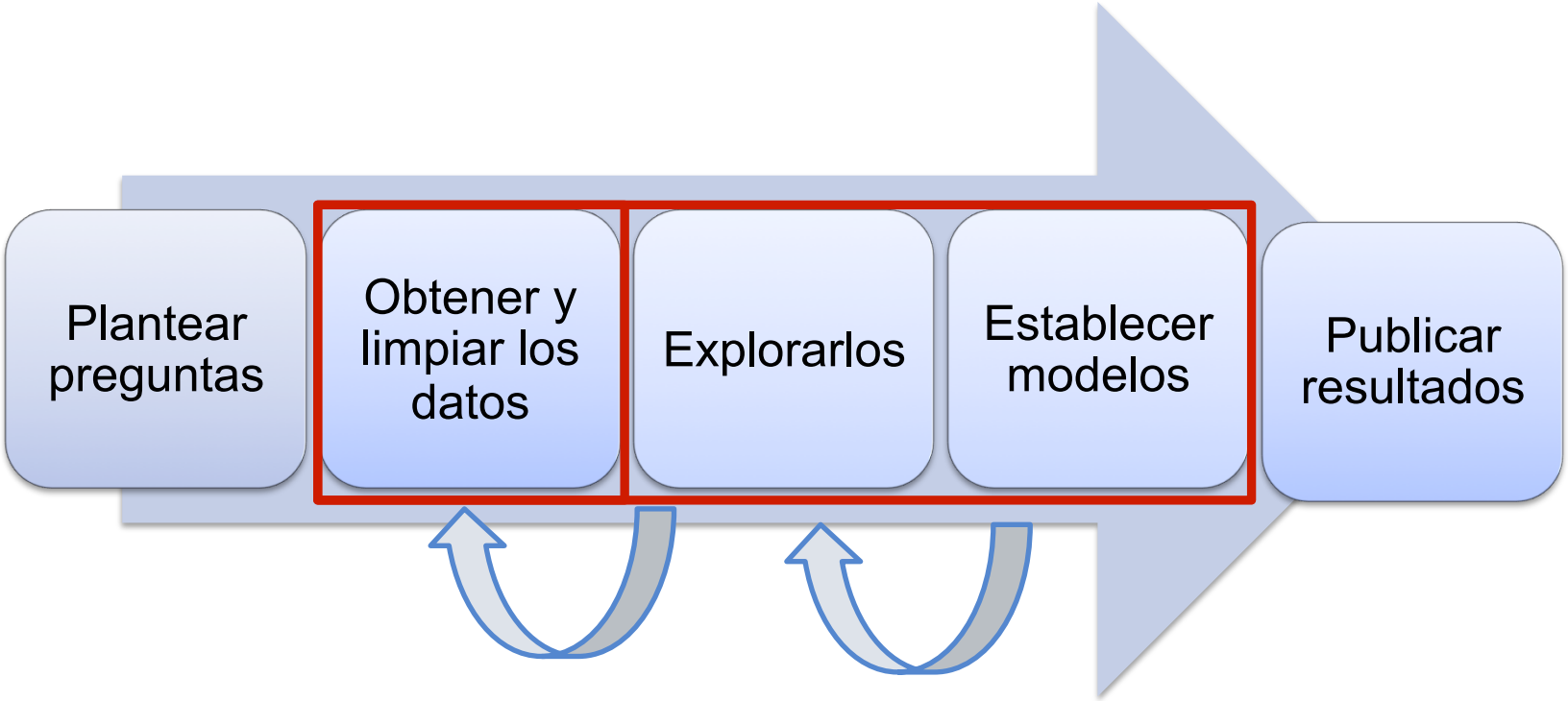
- Pocos datos
- Análisis simple
- No esperamos repetir el análisis

Paquetes estadísticos (SPSS,SAS, Stata,R)

- Tamaño mediano de datos
- Los datos encajan bien con el paquete estadístico

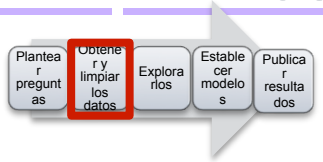
¿Qué hace un científico de datos?

¿Qué hace un científico de datos?

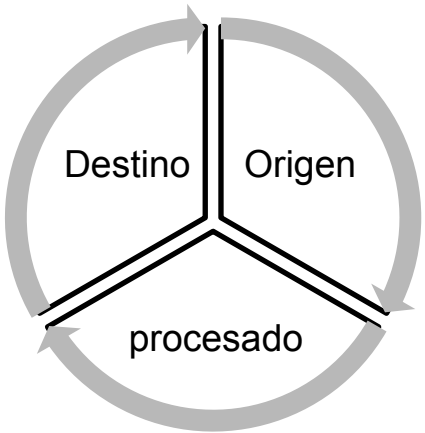


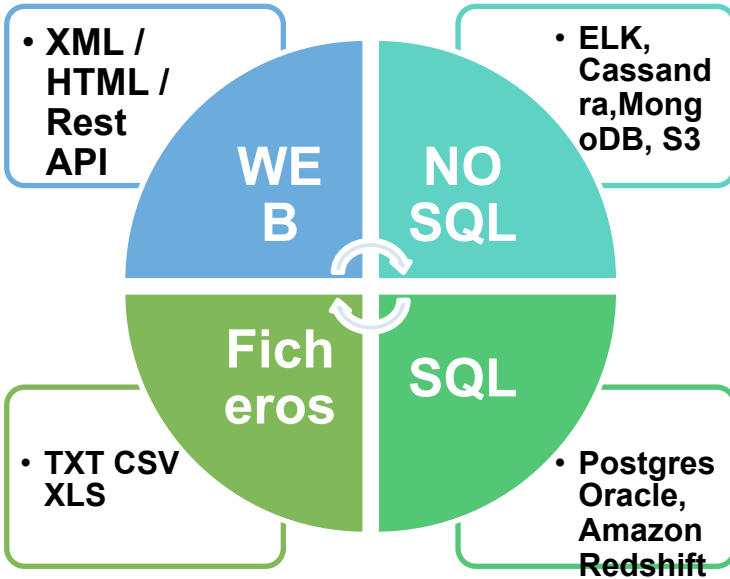
OBTENER y LIMPIAR DATOS

Etapas del proceso

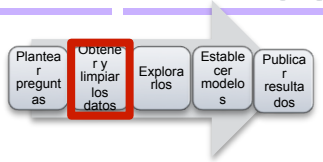


Datos en crudo → Proceso → Datos Procesados



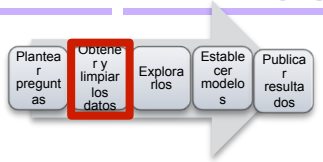


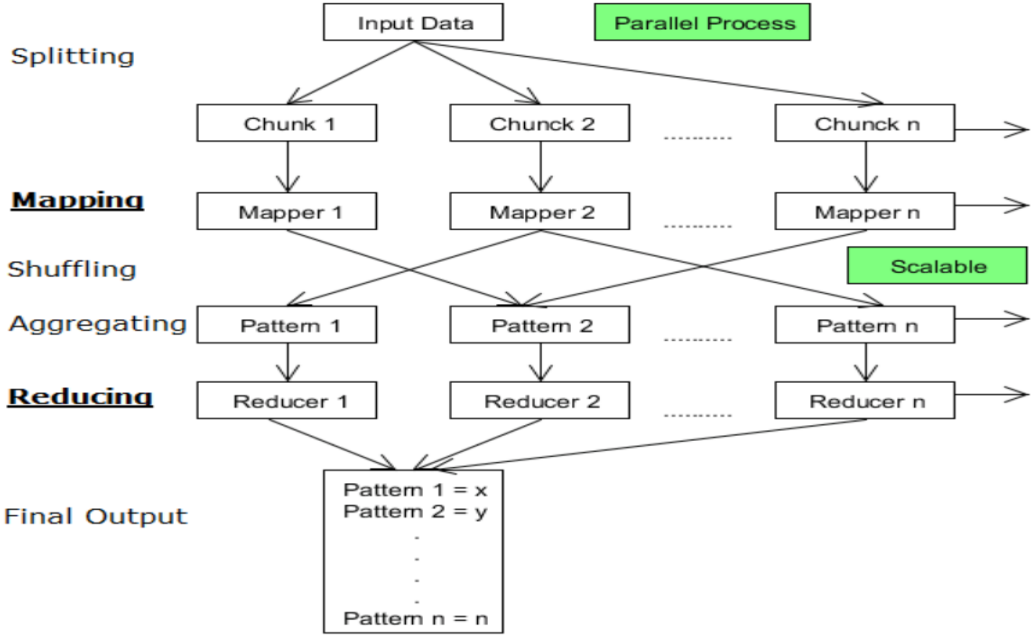
Problemas a resolver…



- Datos no estructurados
- Tratamientos de datos nulos
- Fechas, horas, zonas horarias
- Relacionar/Link de datos.
- Integración de datos de diferentes fuentes
- Volumen/Variedad de los datos

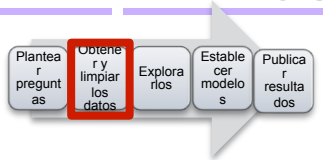
Tratando con un gran volumen de datos: Map-Reduce, Hadoop, Spark





OBTENER y LIMPIAR DATOS

Paquetes Python y R



Python

Requests: accesos HTTP, GET/ POST/PUT/DELETE,etc.

BeautifulSoup/ Scrapy : webscraping gestionar XML / HTML, búsquedas

Datetime, pytz: fechas y zonas horarias.

Re: expresiones regulares.
Pattern: Búsquedas Twitter/ Wikipedia/Google

Hadoop Streaming, mrjob, PySpark, rohive

R

httr-XML: webscraping

Jsonlite: gestionar documentos JSON

Stringr: Manejo de cadenas.

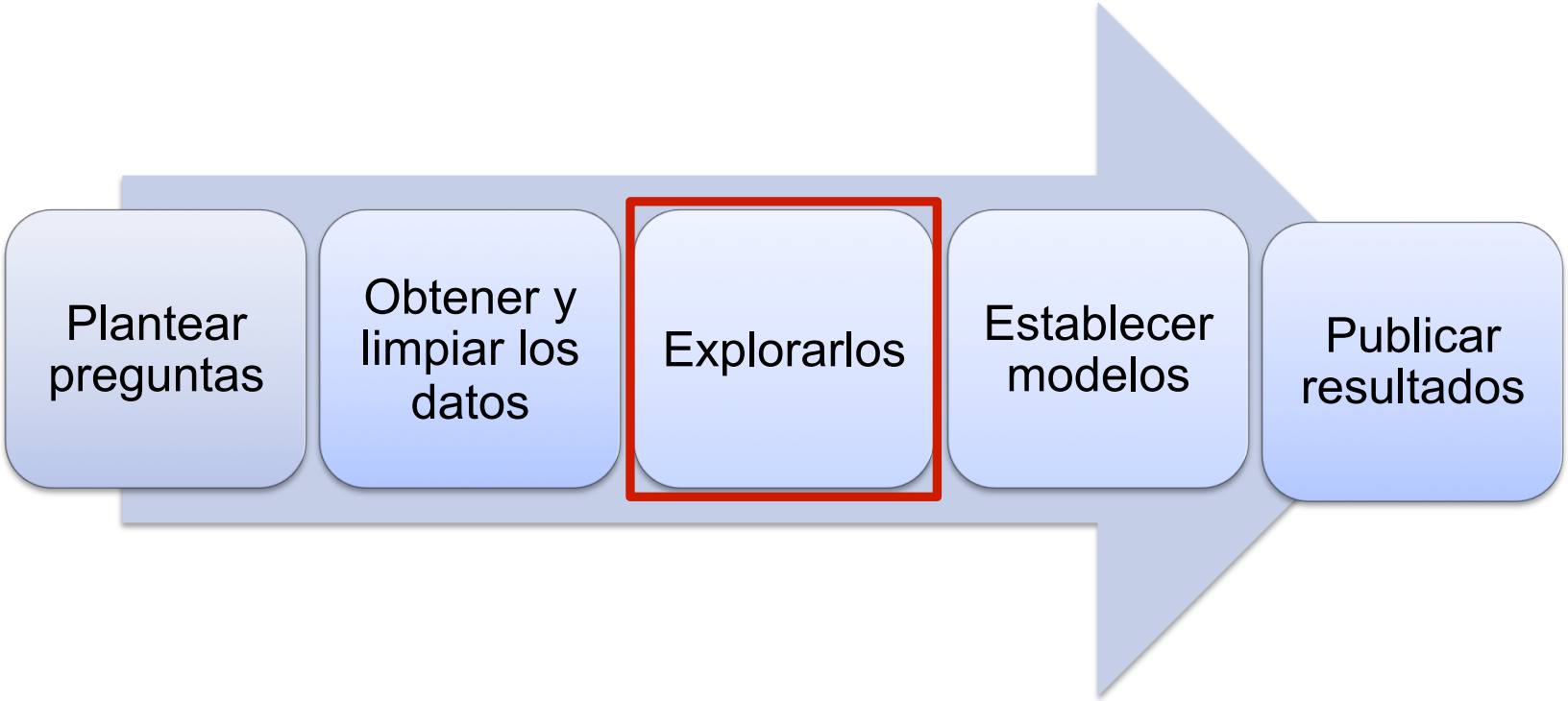
Tidyr: para limpiar datos

Lubridate: Fechas y horas

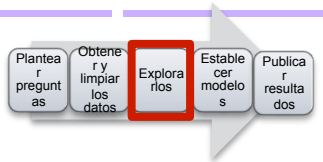
Rmysql, PostgreQL, MySQL , Minitab/S/SAS/SPSS/Stat/Systat RMongo/rmongodb

HadoopStreaming, SparkR, Rhive

1. ¿Qué hace un científico de datos?

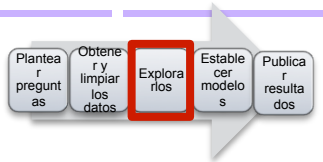


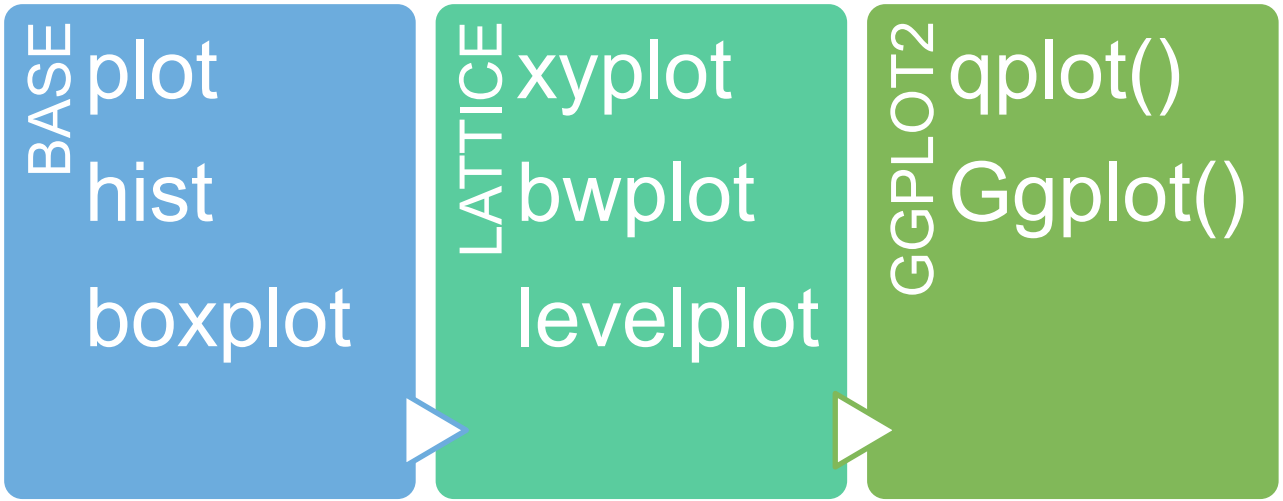
¿Cómo exploramos los datos?



- Seleccionar, buscar, filtrar, agrupar, sumar, ordenar, hacer rangos
- Cálculos estadísticos, operaciones algebraicas.
- Gráficos: Multivariados (1D/2D/3D). Espaciales. Temporales
- Grupos de datos, jerarquías.
- Datos en red
- Observar / Comparar distribuciones / Relacionar datos y patrones

Gráficos en R

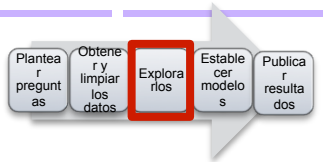




https://www.r-graph-gallery.com/

Explorar los datos

Gráficos en R

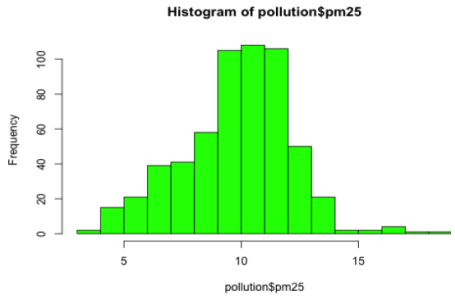



```
summary(pollution$pm25)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3.38    8.55   10.00    9.84   11.40   18.40
```

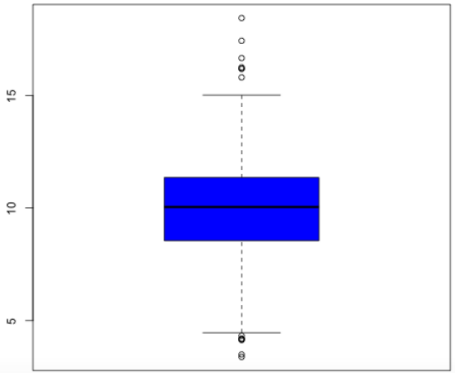
Histogram

```
hist(pollution$pm25, col = "green")
```



Boxplot

```
boxplot(pollution$pm25, col = "blue")
```

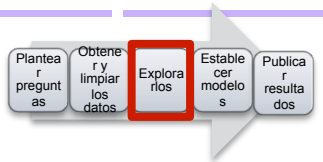


Density

Violin

Explorar los datos

Gráficos en R

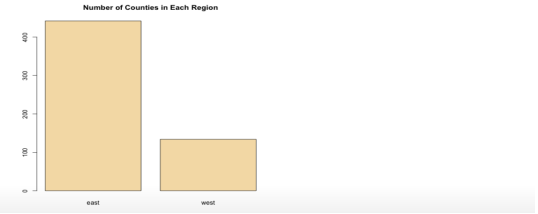


Scatter

Heatmap

Barplot

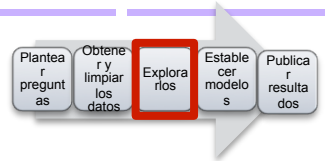
```
barplot(table(pollution$region), col = "wheat", main = "Number of Counties in Each Region")
```



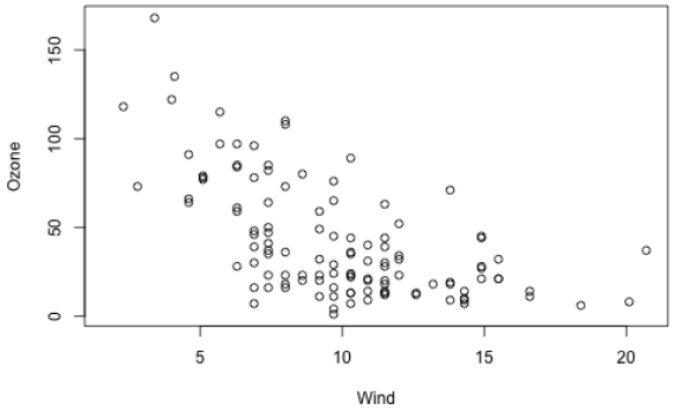
Background Map

Explorar los datos

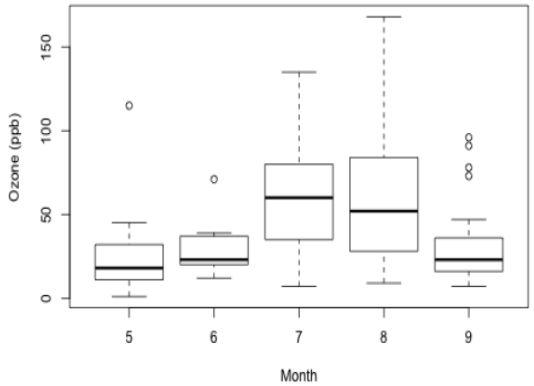
Gráficos en R



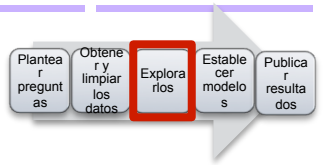

```
library(datasets)
with(airquality, plot(Wind, Ozone))
```



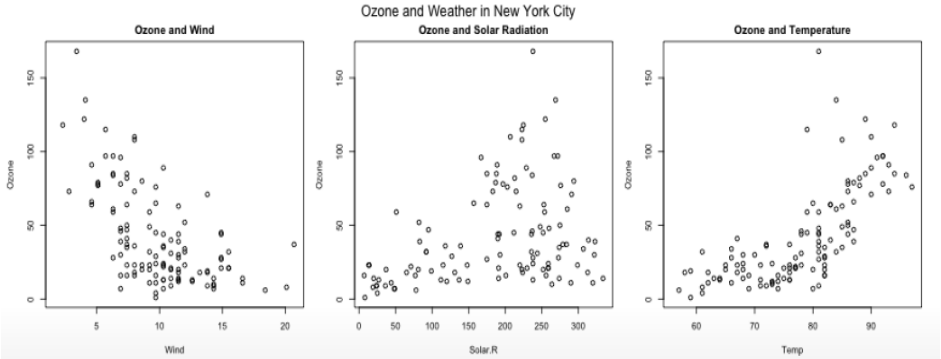

```
library(datasets)
airquality <- transform(airquality, Month = factor(Month))
boxplot(Ozone ~ Month, airquality, xlab = "Month", ylab = "Ozone (ppb)")
```



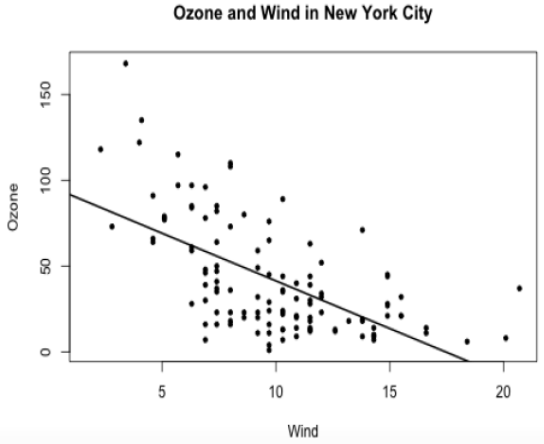
Gráficos en R




```
par(mfrow = c(1, 3), mar = c(4, 4, 2, 1), oma = c(0, 0, 2, 0))
with(airquality, {
    plot(Wind, Ozone, main = "Ozone and Wind")
    plot(Solar.R, Ozone, main = "Ozone and Solar Radiation")
    plot(Temp, Ozone, main = "Ozone and Temperature")
    mtext("Ozone and Weather in New York City", outer = TRUE)
})
```

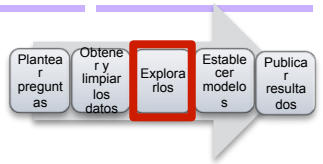



```
with(airquality, plot(Wind, Ozone, main = "Ozone and Wind in New York City",
    pch = 20))
model <- lm(Ozone ~ Wind, airquality)
abline(model, lwd = 2)
```

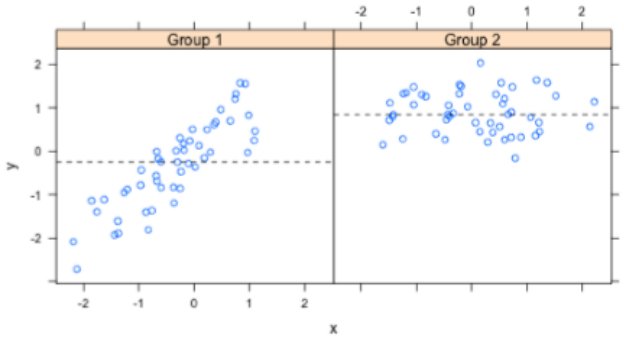


Explorar los datos

Gráficos en R

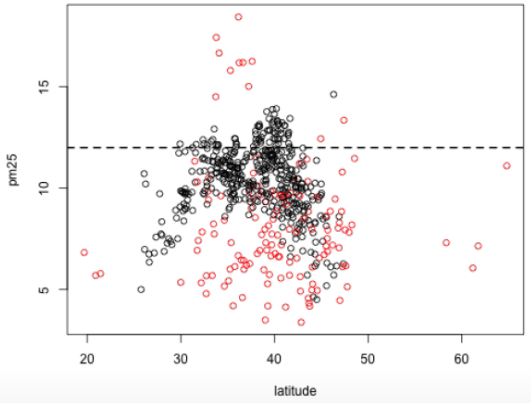



```
## Custom panel function
xyplot(y ~ x | f, panel = function(x, y, ...) {
    panel.xyplot(x, y, ...)  ## First call the default panel function for 'xyplot'
    panel.abline(h = median(y), lty = 2)  ## Add a horizontal line at the median
})
```

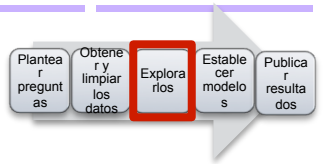


Scatterplot - Using Color

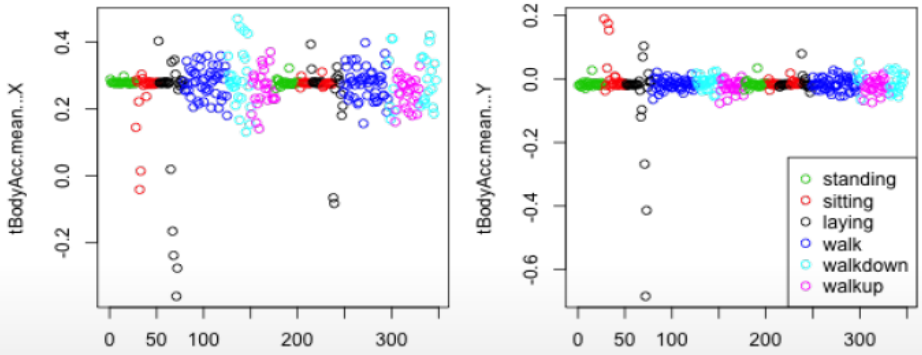
```
with(pollution, plot(latitude, pm25, col = region))
abline(h = 12, lwd = 2, lty = 2)
```

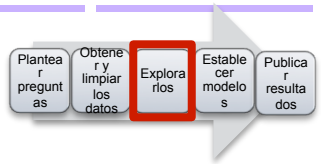


Gráficos en R



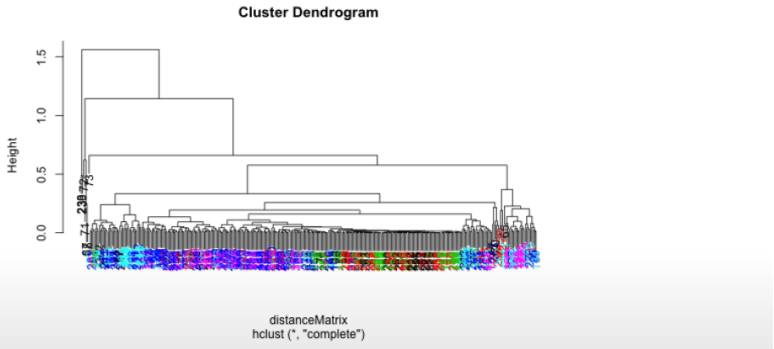

```
par(mfrow = c(1, 2), mar = c(5, 4, 1, 1))
samsungData <- transform(samsungData, activity = factor(activity))
sub1 <- subset(samsungData, subject == 1)
plot(sub1[, 1], col = sub1$activity, ylab = names(sub1)[1])
plot(sub1[, 2], col = sub1$activity, ylab = names(sub1)[2])
legend("bottomright", legend = unique(sub1$activity), col = unique(sub1$activity),
    pch = 1)
```





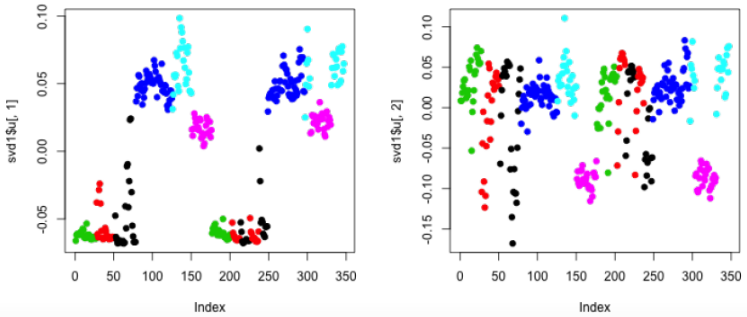
Clustering based just on average acceleration

```
source("myplclust.R")
distanceMatrix <- dist(sub1[, 1:3])
hclustering <- hclust(distanceMatrix)
myplclust(hclustering, lab.col = unclass(sub1$activity))
```



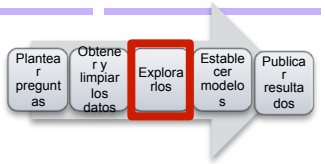
Singular Value Decomposition

```
svd1 = svd(scale(sub1[, -c(562, 563)]))
par(mfrow = c(1, 2))
plot(svd1$u[, 1], col = sub1$activity, pch = 19)
plot(svd1$u[, 2], col = sub1$activity, pch = 19)
```



Explorar los datos

Paquetes Python y R



Python

Panda: Utilizado para transformación de datos. Agrupados, joins, series de tiempo.

Numpy / Scipy: funciones numéricas, arreglos multidimensionales.

Matplotlib: gráficos 2D, de líneas, histogramas, torta, etc.

Statsmodel: exploración estadística.

Seaborn: heat maps, time series, y violin plots.

R

Dplyr: Permite tratar datos, con modo “sql”: select, filter, arrange, summarize, group by, mutate.

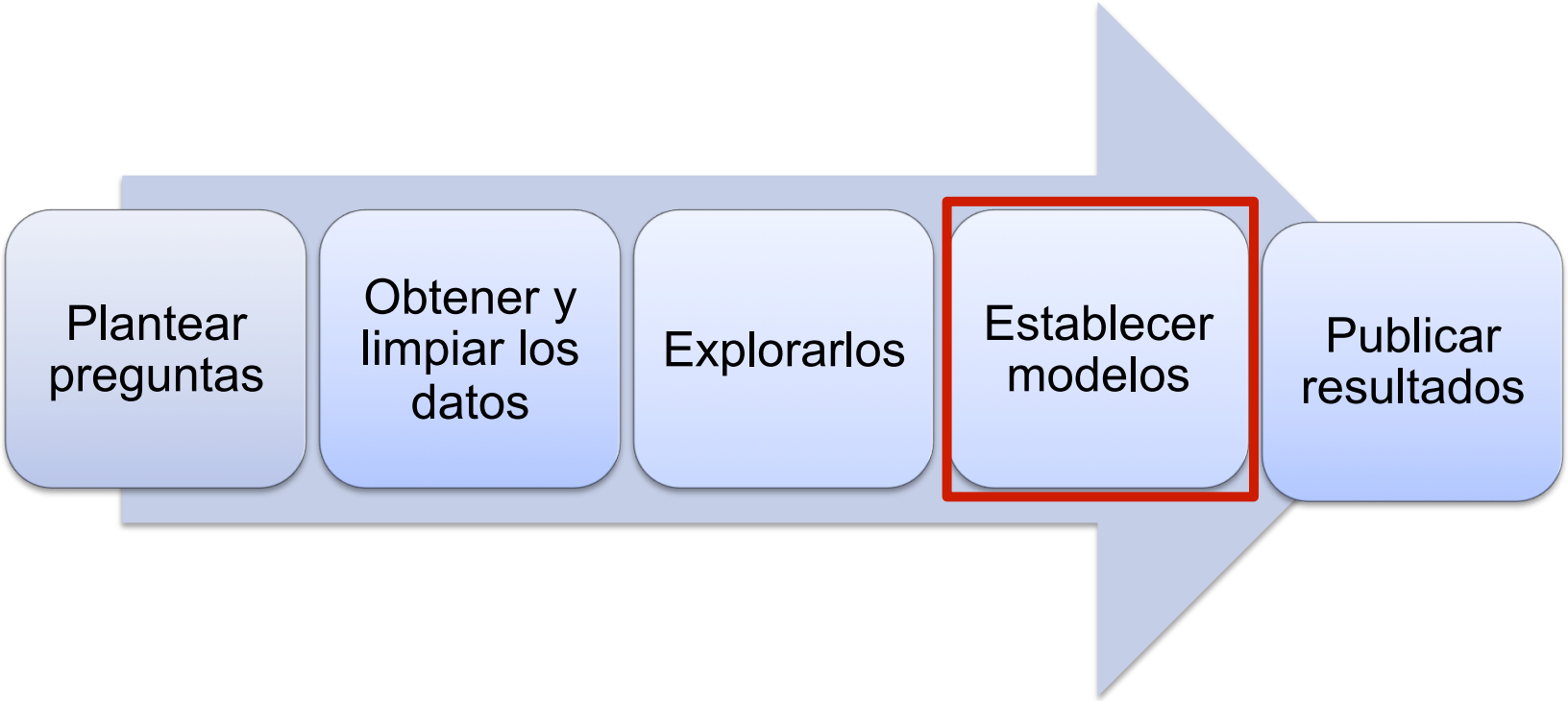
Base, lattice, ggplot2

Stringr: manejo de cadenas.

lubridate: fechas / horas.

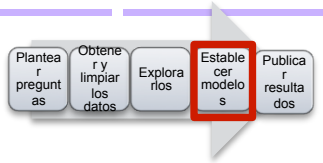
Xda: análisis de variables dependientes vs independientes, stats, gráficos, reemplazo de nulos.

¿Qué hace un científico de datos?



ESTABLECER MODELOS

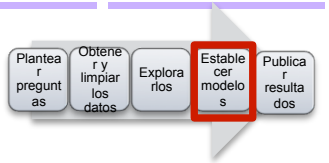
Métodos de *Machine Learning*



- Aprendizaje No supervisados
  - Clustering
  - Análisis de componentes principales
  - Reglas asociativas
- Aprendizaje Supervisado
  - Clasificación
  - Regresión

- K-Nearest neighbor
- Support Vector Machines
- Decisión trees
- Ensemble methods
- Random forests
- Neural networks

Paquetes Python y R



Python

Scikit-learn: supervisados y no supervisados.

TensorFlow: redes neuronales, reconocimiento de imágenes.

Theano: evalúa exp.matemáticas Optimizaciones: uso de GPU

PyBrain: redes neuronales, aprendizaje no supervisado.

R

Caret: Clasificación y regresión

Ranger / Randomforest: modelo muy usado para clasificar.

E1071: Support Vector Machines **con** gpu.

Xgboost: modelos lineales, árboles.

Rpart: particionado para clasificación.

Glmnet: modelos lineales

Alumni

alumni.uoc.edu

Gracias por su atención

AlumniUOC
@UOCalumni