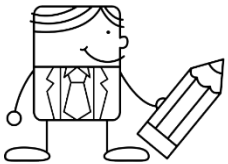
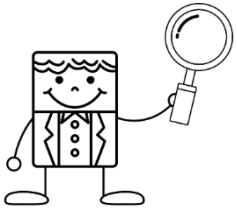
Introducción a Data Science en R:

Aprendizaje automático: supervisado y no supervisado

- Dentro del tema encontrarás los epígrafes de contenido, con subepígrafes para una mejor comprensión.
- Todos los temas han sido elaborados siguiendo el criterio de facilidad de exposición.
- Los contenidos incluyen, en este sentido, cinco tipo de llamadas de atención que agilizan y facilitan la comprensión de los mismos:

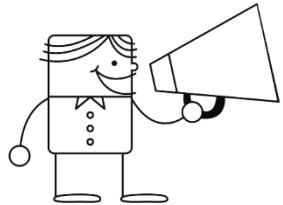


Ejemplo

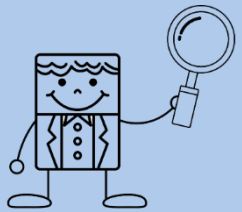


Tenga en cuenta que….

Recuerde



Definición



Tenga en cuenta que este curso tiene un componente práctico muy fuerte. Es labor del alumno practicar todos los ejemplos y ejercicios planteados.

"A programar se aprende programando"

1 Aprendizaje No Supervisado

Aprendizaje No Supervisado

- Se trata de herramientas cuyo objetivo es **entender los datos**, sin una variable objetivo (o supervisor), y **organizarlos en grupos de manera natural**.

- Mientras que, recuérdese, que, por otro lado, el aprendizaje supervisado consiste en predecir o estimar una variable objetivo (o respuesta) desde varios inputs (predictores).

- El **Clustering** es una las técnicas más utilizadas en Aprendizaje No Supervisado, otra es el PCA (Principal Component Analysis).

- En general, es difícil determinar el número de clusters óptimo, o grupos naturales en los que se organizan los datos.

1.1 Cluster Analysis

Outline del Clustering

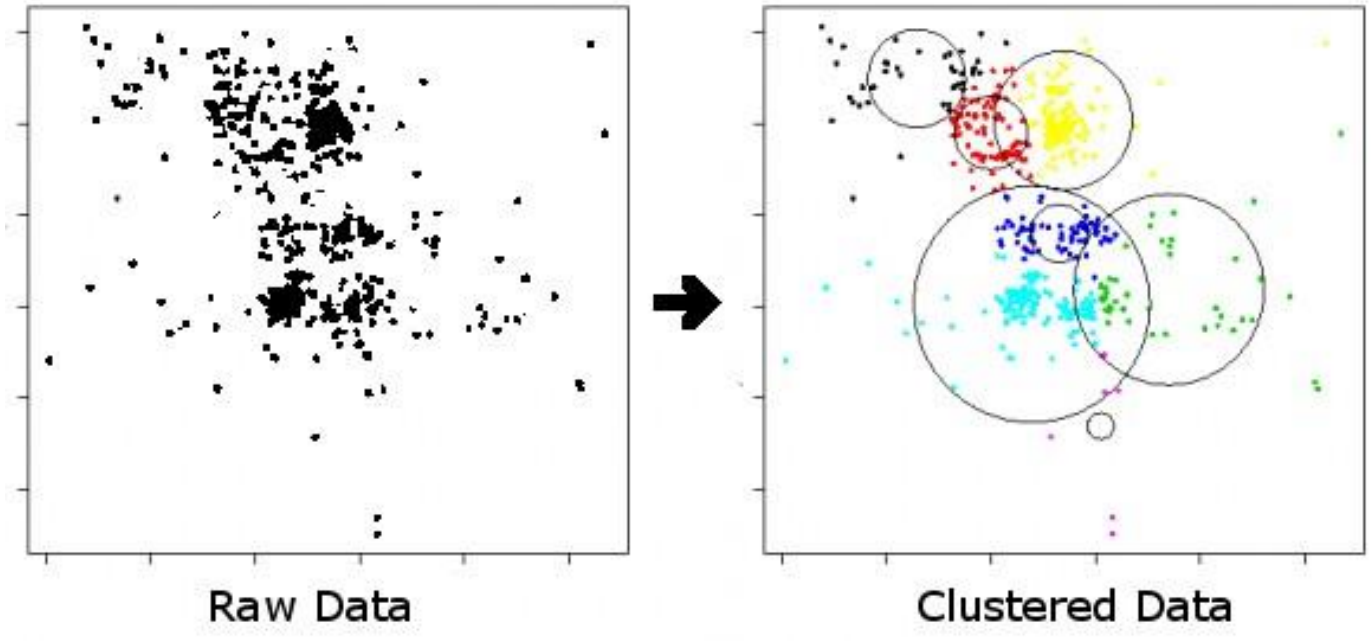
Clasificar las observaciones de una matriz de datos ***X*** (p.e. un DataFrame de R) en **grupos homogéneos**

- Las observaciones dentro de cada grupo deben ser similares.
- Las observaciones de diferentes grupos deben ser diferentes.
- Se mide la **similaridad** o **proximidad** a través de **distancias**.
- No conocemos *a priori* el grupo al que pertenecen las observaciones.
- Tampoco conocemos *a priori* el número de grupos.

Aplicaciones del Clustering

- Marketing: Organizar los clientes en diferentes perfiles (según su consumo, edad, etc.) para ofrecer campañas de publicidad dirigidas.
- Finanzas: Clasificar compañías según su rendimiento en los mercados financieros.
- Salud: Organizar pacientes en grupos de tratamiento.
- Seguros: Identificar grupos de asegurados con altos costes de siniestros.
- Redes Sociales: Identificar comunidades.
- Genética: Selección genética.
- Otras: Sistemas de recomendación, detección de anomalías, análisis de imagen, grupos de resultados de búsqueda, etc.

Idea principal del Clustering



Ideas Básicas

- El Clustering estudia datos para los cuales el número de grupos es desconocido e indefinido.
- Necesitamos poner el foco en distancias **intra-cluster**, para incrementar la similaridad
  - Cómo de cerca están los datos unos de otros
  - Se denomina coloquialmente distancia o medida de similaridad
- Y también hay que tener en cuenta la distancia **inter-cluster**, para disminuir la similaridad
  - Cómo de cerca están los *clusters* entre sí
  - Se denomina comúnmente la función *linkage*
- Por tanto, la única información que usa el clustering son **similaridades**.

Ejemplo

ID	Gender	Age	Salary	Balance
1	F	27	21000	550
2	M	51	64000	900
3	M	53	75000	825
4	F	32	55000	1100
5	M	45	50000	875
6	F	37	45000	650

¿Qué clientes son más similares?

Ejemplo

	Term1	Term2	Term3	Term4	Term5	Term6
Doc1	0	2	0	0	1	0
Doc2	3	5	4	3	0	0
Doc3	3	0	0	0	3	4
Doc4	0	0	0	3	0	4
Doc5	2	1	2	3	0	1
Doc6	1	4	2	1	2	0

¿Qué documentos son más similares?

Donde Term1, Term2, … son variables que expresan la frecuencia de aparición de diferentes términos en los diferentes documentos (Doc1, Doc2,…).

Cluster Analysis: Variables/Features

- Las variables o *features* que incluímos en el análisis pueden tener gran impacto en la solución.
- En este punto es necesario recurrir al conocimiento de la posible aplicación del analista, su creatividad y *expertise*.
- Un buen análisis exploratorio suele ser de ayuda. Por ejemplo: en un dataset de banca, las variables asociadas con los atributos socio-económicos (como income/balance/risk) se pueden usar para la segmentación; y el resto de variables, como las características de los clientes (profesión, región en la que viven, etc) pueden ayudar a describir los grupos que devuelve el clustering.

Cluster Analysis: Herramientas

- **Métodos de Partición**: Dividen las observaciones en un número de grupos pre-especificado.

  Objetivo: Agrupar observaciones similares basándose en alguna **distancia (similaridad)** entre observaciones:
  - **K -means**: Los objetos dentro de cada cluster se encuentran lo más cerca posible entre sí, y lo más lejos posible de los objetos de otros clusters. Cada cluster se caracteriza por un centroide.
  - **Métodos basados en modelos**: Mezclas de distribuciones estadísticas.

- **Métodos Jerárquicos**: Comienzan con clústeres con una sola observación y unen los clústeres en pasos iterativos posteriores.

- Otros modelos se basan en técnicas de Machine Learning.

1.2 K- Means

Cluster Analysis: Métodos de Partición

- Es necesario fijar previamente el número de grupos, *K.*
- Al final del algoritmo de clustering cada observación va a uno de estos grupos.
- La herramienta más popular de partición: *K -means*
  - La métrica de distancia es la **Euclídea** (la distancia natural en nuestro espacio tridimensional)
  - Proporciona una solución razonable
  - Es muy rápido.

Cluster Analysis: K -means

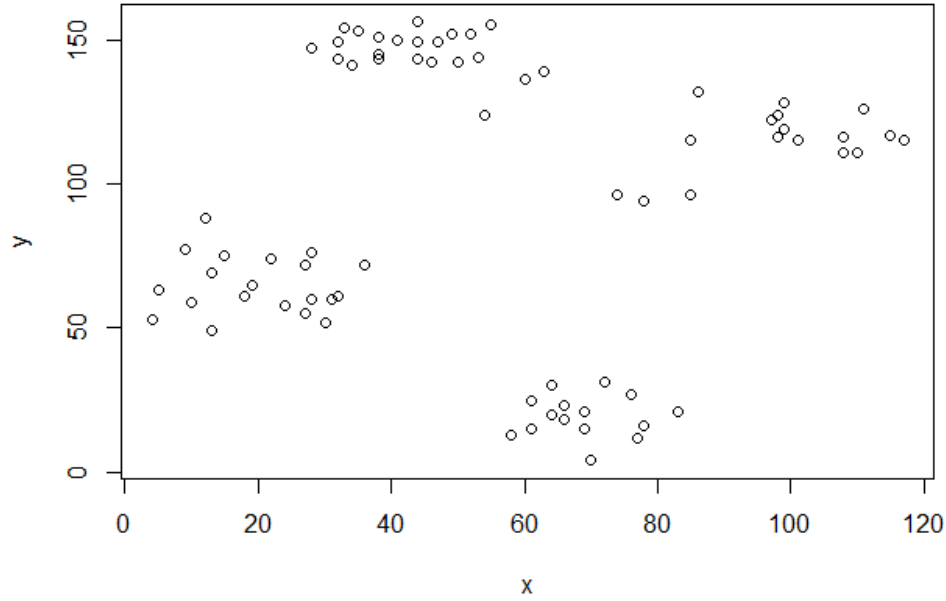
- Asigna aleatoriamente cada observación a uno de los K grupos.
- Calcula las K medias de cada muestra (centroide) de las observaciones de cada grupo.
- Asigna cada observación al grupo con la media más cercana (usando la norma euclídea)
- Repite los dos pasos anteriores hasta que no cambian los grupos.

Ilustración animada: Link

Ilustración animada: Link

Cluster Analysis: Ejemplo

Dataset sencillo: 75 puntos en 2D

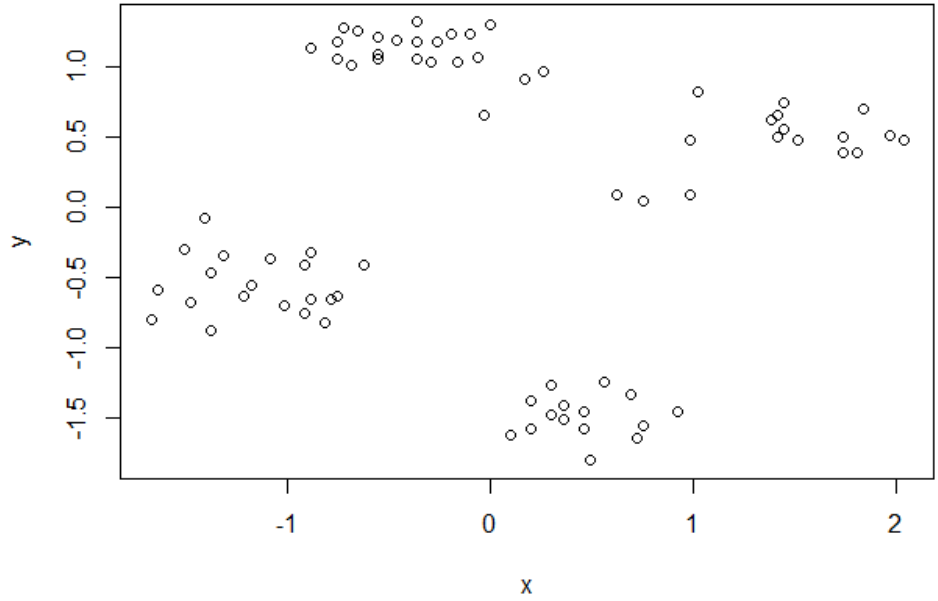



```
### Partition methods
install.packages("cluster")
library(cluster)

# easy data set, just for slides
ruspini <- ruspini[sample(1:nrow(ruspini)),]

plot(ruspini,main="")
```

Se utiliza la librería **`cluster`** *para realizar el análisis de clustering. Además del dataset* **`ruspini`**, *contenido en la misma.*

Cluster Analysis: Ejemplo

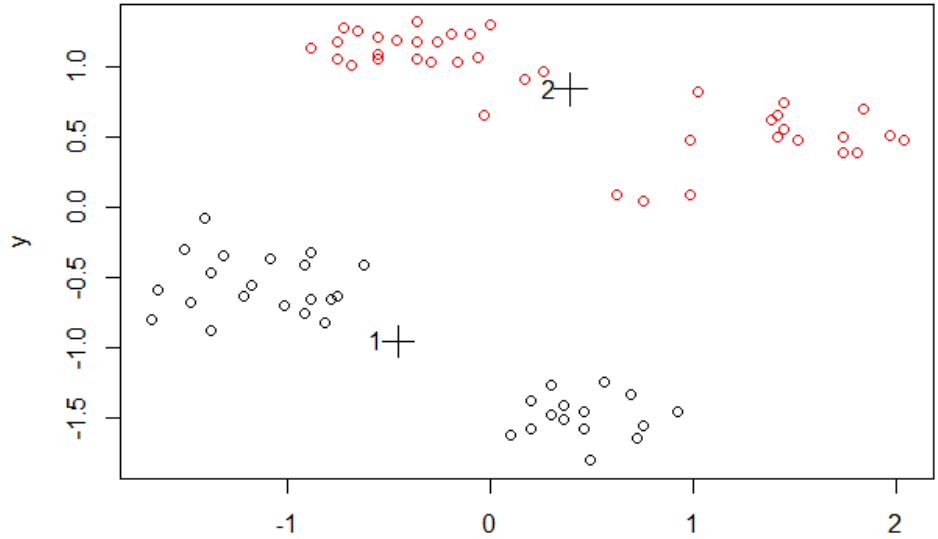
Estandarizar (Restar la media y dividir por la desviación estándar) las variables




```
# Prepare Data
Ruspini <- scale(ruspini) # standardize variables
# This may prevent one attribute with a large range to dominate
# the others for the distance calculation

plot(Ruspini,main="")
```

Cluster Analysis: Ejemplo

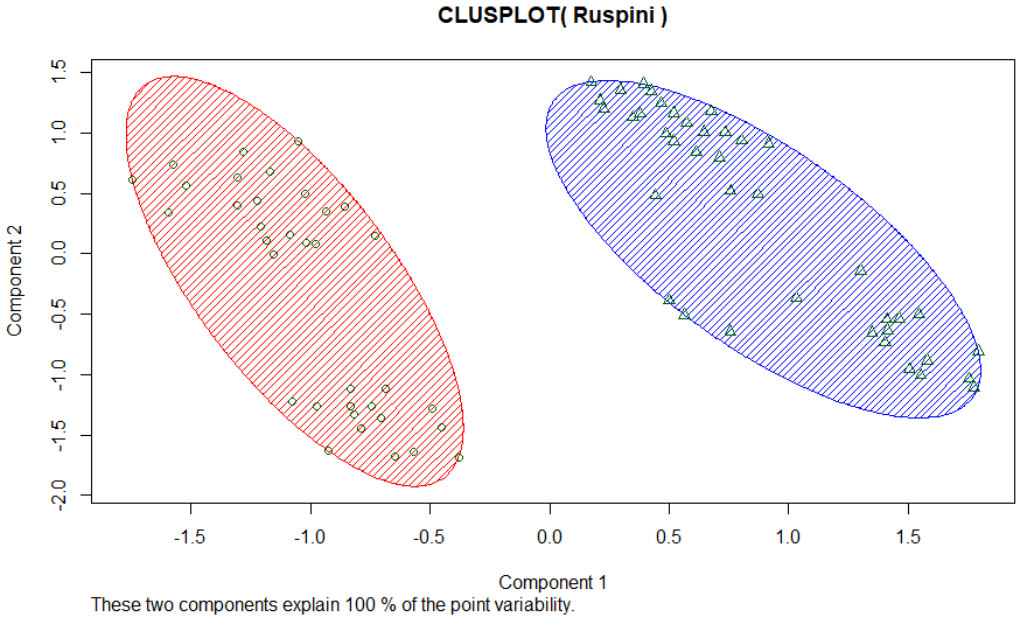
Consideremos que hay dos grupos.




```
### K-Means Cluster Analysis

# Assume first k=2 clusters and Euclidean distances
# Run the algorithm 20 times with random initialized centroids
# The best result is returned
fit.kmeans <- kmeans(Ruspini, centers=2, nstart=20) # 2 cluster solution
fit.kmeans

plot(Ruspini, col=fit.kmeans$cluster)
points(fit.kmeans$centers, pch=3, cex=2) # this adds the centroids
text(fit.kmeans$centers, labels=1:2, pos=2) # this adds the cluster ID
```

Cluster Analysis: Ejemplo

En alta dimensionalidad, es útil dibujar los clusters en las primeras componentes principales

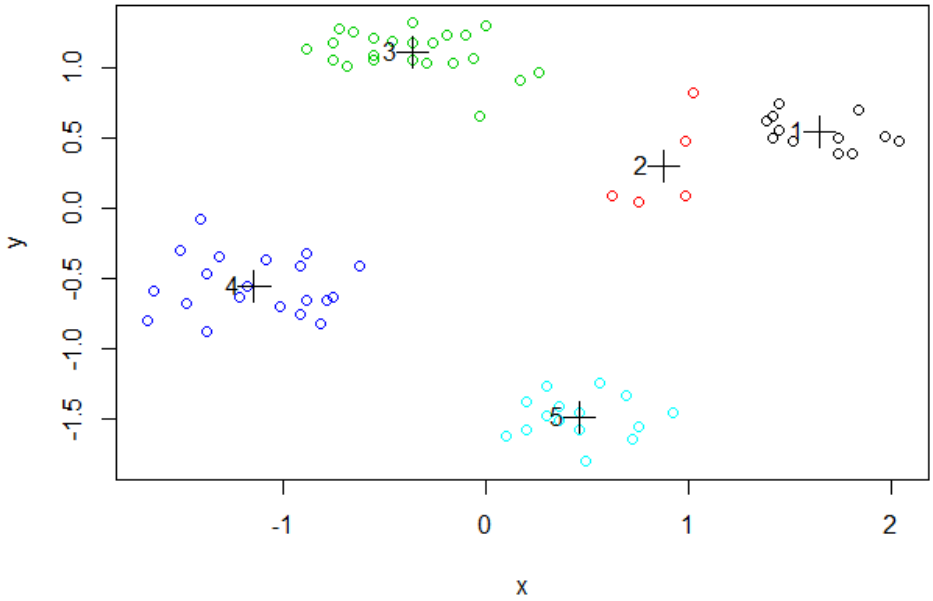



```
# plot the points using the first principal components (useful in dimension>2)
clusplot(Ruspini, fit.kmeans$cluster,color=TRUE,shade=TRUE,lines=0)
```

La **descomposición en componentes principales** es un concepto complicado que no se describirá en detalle pero baste decir que **reducen la dimensionalidad del dataset**, y son combinaciones de las variables que **buscan explicar la mayor proporción de variabilidad** del dataset. En el caso que nos ocupa las dos primeras componentes principales son idénticas a las dos variables del problema.

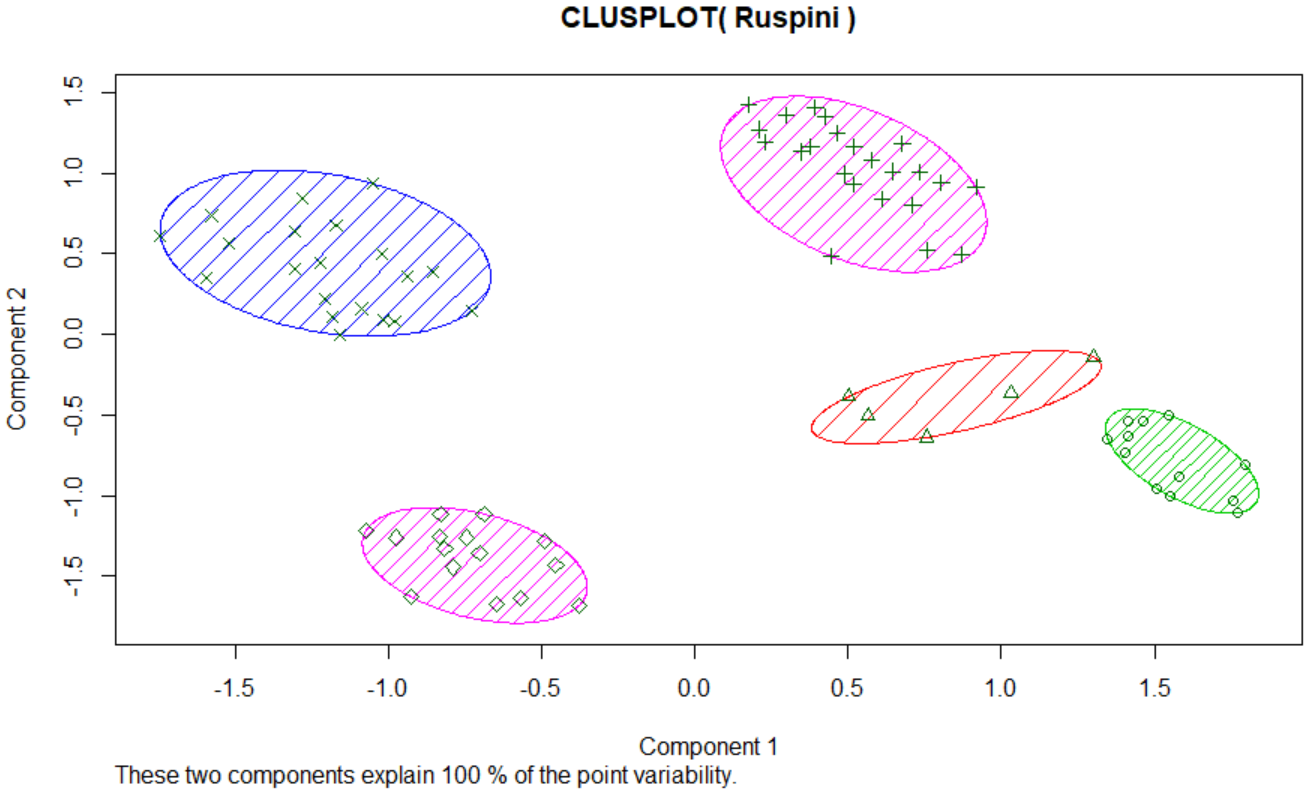
Cluster Analysis: Ejemplo

Consideremos ahora que existen 5 grupos




```
# Assume now k=5 clusters and Euclidean distances
fit.kmeans <- kmeans(Ruspini, centers=5, nstart=20) # 5 cluster solution
fit.kmeans

plot(Ruspini, col=fit.kmeans$cluster)
points(fit.kmeans$centers, pch=3, cex=2) # this adds the centroids
text(fit.kmeans$centers, labels=1:5, pos=2) # this adds the cluster ID
```

Cluster Analysis: Ejemplo




```
# plot the points using the first principal components (useful in dimension>2)
clusplot(Ruspini, fit.kmeans$cluster,color=TRUE,shade=TRUE,lines=0)
```

*Practica el Ejemplo **12_clustering.R***

¡No pierdas la pista a los ejemplos en la carpeta correspondiente!

2 Aprendizaje Supervisado

Aprendizaje Supervisado

- Busca **predecir o estimar una variable objetivo** (o respuesta) o variable dependiente, utilizando el **resto de variables** como input o **variables independientes** (denominadas variables explicativas o predictores).

- Para ello utiliza la variable respuesta conocida en el dataset de entrenamiento como supervisor o referencia en el ajuste.

- El modelo final que emula el comportamiento de la variable objetivo en función de las variables independientes debe ser suficientemente generalizable, ya que llegado el caso, se utilizará para predecir la variable respuesta.

- Esto último se consigue evitando el sobreajuste, comportamiento indeseado cuando un modelo ajusta demasiado bien los datos de entrenamiento pero no da predicciones buenas cuando se utiliza en datos de entrada nuevos.

- Si la **variable respuesta** es **continua** nos encontramos ante un problema de **regresión**, **si es discreta** ante un problema de **clasificación**.

2.1 Regresión

Regresión Lineal

El modelo más sencillo de regresión es la **regresión lineal**

- Expresa la variable dependiente como combinación lineal de las variables independientes o predictores más un término constante.

- La ecuación fundamental de la regresión lineal se expresa como:

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

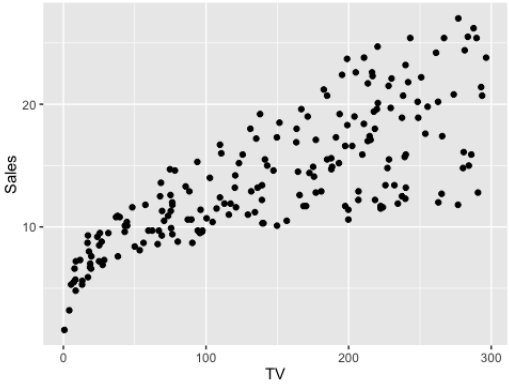
  donde Y es la variable objetivo y $X_1, \ldots, X_p$ son las variables independientes, todas de longitud N, que es el número de observaciones (dataset de N filas y p columnas)

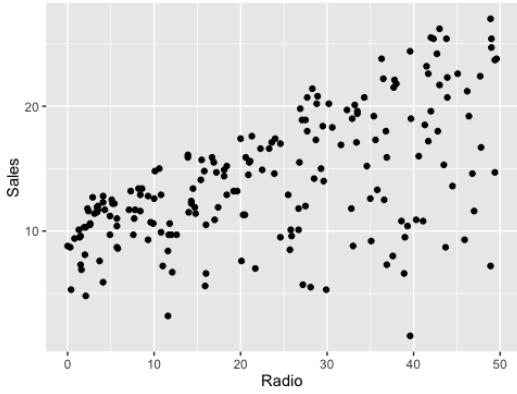
- Los coeficientes $\beta_1, \ldots, \beta_p$ son los **coeficientes de la regresión** y expresan cuánto crece la variable respuesta cuando aumenta una unidad la variable predictora a la que multiplica, si se mantienen constantes las demás.

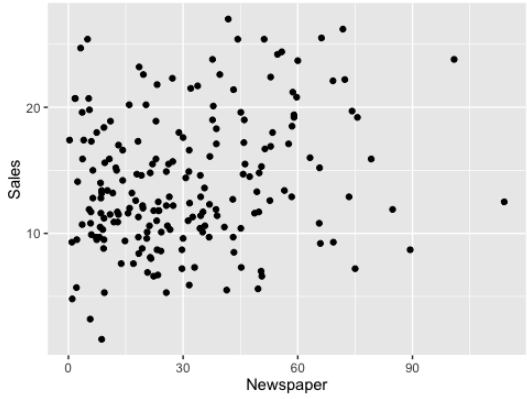
- El término β_0 es el término constante o **intercept**.

Regresión Lineal: Ejemplo

Queremos aconsejar a una compañía sobre cómo mejorar las ventas de un determinado producto (en miles de unidades). Para conseguirlo, nos proporcionan un set de datos que contiene las ventas del producto en 200 mercados diferentes, junto con el presupuesto de publicidad en televisión, radio y periódicos en cada uno de tales mercados (en miles de dólares).







Naturalmente, nuestro cliente no tiene la capacidad de incrementar las ventas de su producto directamente, pero sí tiene la capacidad de decidir qué presupuesto dedicar a cada uno de los canales de publicidad

Regresión Lineal: Ejemplo

El dataset se puede cargar desde **advertising.csv**

```
# Load dataset
advertising <- read.csv('advertising.csv', sep = ';', header = T, fileEncoding = 'utf-8')
```

El primer modelo consiste en una **Regresión Lineal Simple** de las Ventas (**Sales**) como variable objetivo frente al canal de publicidad de Televisión (**TV**) como único predictor.

```
# Single variable regression
lm_fit_sales_TV <- lm(Sales ~ TV, data = advertising)

lm_fit_sales_TV
```

```
> lm_fit_sales_TV

Call:
lm(formula = Sales ~ TV, data = advertising)

Coefficients:
(Intercept)           TV
    7.03259      0.04754
```

El modelo resultante tiene como **ecuación**:

$$Sales = 7.03259 + 0.004754 \cdot TV$$

La instrucción para ajustar un modelo lineal en R es **`lm`**:

```
lm(formula, data, subset, weights, ...)
```

Donde

- **`formula`** es una descripción simbólica del modelo que se pretende ajustar.
- **`data`** es un *data frame* que contiene los datos con los que se pretende ajustar el modelo.
- **`subset`** es un vector opcional que especifica el subconjunto de observaciones que se van a usar en el proceso de ajuste.
- **`weights`** es un vector opcional de pesos que se van a usar en el proceso de ajuste del modelo. Por defecto **`weights=NULL`**

Regresión Lineal: Ejemplo

Un modelo más sofisticado puede tener en cuenta los tres canales de publicidad (**TV, Radio, Newspaper**) para describir las ventas (**Sales**) a través de una **Regresión Lineal Múltiple**.

```
# Multiple Linear Regression
lm_fit_sales_all <- lm(Sales ~ TV + Radio + Newspaper, data = advertising)

lm_fit_sales_all
```

```
> lm_fit_sales_all

Call:
lm(formula = Sales ~ TV + Radio + Newspaper, data = advertising)

Coefficients:
(Intercept)           TV        Radio    Newspaper
   2.938889     0.045765     0.188530    -0.001037
```

El modelo resultante tiene como **ecuación**:

$$Sales = 2.938889 + 0.045765 \cdot TV + 0.188530 \cdot Radio - 0.001037 \cdot Newspaper$$

Regresión Lineal: Ejemplo

La función `summary` devuelve información avanzada sobre el modelo, no solo los coeficientes.

```
> summary(lm_fit_sales_all)

Call:
lm(formula = Sales ~ TV + Radio + Newspaper, data = advertising)

Residuals:
    Min      1Q  Median      3Q     Max
-8.8277 -0.8908  0.2418  1.1893  2.8292

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.938889   0.311908   9.422   <2e-16 ***
TV           0.045765   0.001395  32.809   <2e-16 ***
Radio        0.188530   0.008611  21.893   <2e-16 ***
Newspaper   -0.001037   0.005871  -0.177     0.86
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.686 on 196 degrees of freedom
Multiple R-squared:  0.8972,    Adjusted R-squared:  0.8956
F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

Estadísticos sobre los **residuos**. Los **residuos** son la diferencia entre los valores de la variable objetivo y la salida del modelo. Cuánto más pequeños y centrados en torno a cero mejor ajuste del modelo a los datos.

Estos asteriscos reflejan un test estadístico de relación entre los predictores y la variable de salida (*p-valor*). Si no hay asteriscos no hay una relación entre la variable predictora y la variable respuesta. En este caso, no se puede decir que **Newspaper** sea una variable significativa en el modelo, no influye en **Sales**.

`Multiple` y **`Adjusted R-squared`** **cercanos a 1** significa que el rendimiento del modelo es bueno.

*Trabaja el Ejemplo **13_regression.R***

¡No pierdas la pista a los ejemplos en la carpeta correspondiente!

2.1 Clasificación

Regresión Logística

El modelo más sencillo de clasificación es la **regresión logística**

- Busca modelizar problemas de clasificación en dos categorías, donde la **variable objetivo** es discreta y tiene solo **dos categorías posibles** (que se codifican por los valores **0 y 1**).
- Es una técnica de clasificación estadística, en la que primero se predice la probabilidad de cada categoría p, y luego la categoría.
- Lo que se modeliza ahora es la cantidad $\frac{p}{1-p}$, denominados los **odds.**
- Un valor pequeño indica una probabilidad baja de que $Y = 1$, mientras que un valor grande indica una alta probabilidad.

Regresión Logística

- Se puede relacionar con un problema de regresión lineal si se modeliza:

$$\text{logit}(p) \equiv \log \frac{p}{1-p} = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

donde p es la probabilidad de pertenencia a cada categoría y $X_1, \ldots, X_p$ son las variables independientes, de longitud N, el número de observaciones (dataset de N filas y p columnas)

- Si incrementamos la variable X_i por una unidad, el $\text{logit}(p)$ crece una cantidad igual a la β_i correspondiente.

- Una vez que tenemos las probabilidades de pertenencia de cada clase se puede hacer, en el supuesto más sencillo, que:

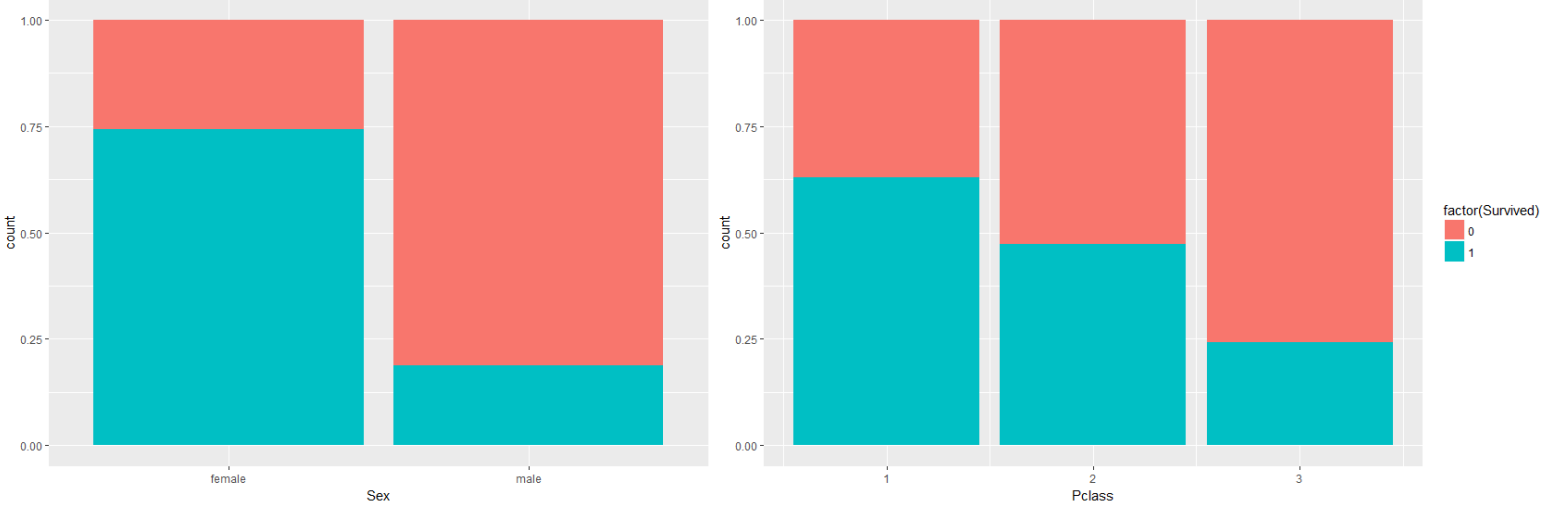
$$Y = \begin{cases} 1 \; si \; p \geq 0.5 \\ 0 \; si \; p < 0.5 \end{cases}$$

Regresión Logística: Ejemplo

Utilizaremos el dataset **titanic.csv** que contiene datos de los pasajeros del Titanic, relacionados con su Sexo (**Sex**), Edad (**Age**), Tarifa de Embarque (**Fare**), Clase de Pasajero (**PClass**), entre otros, y la variable objetivo **Survived** que denota si el pasajero en cuestión sobrevivió o no (**1** ó **0**, resp).

```
titanic <- read.csv('titanic.csv', header = T, stringsAsFactors = F, encoding = 'utf-8')
```

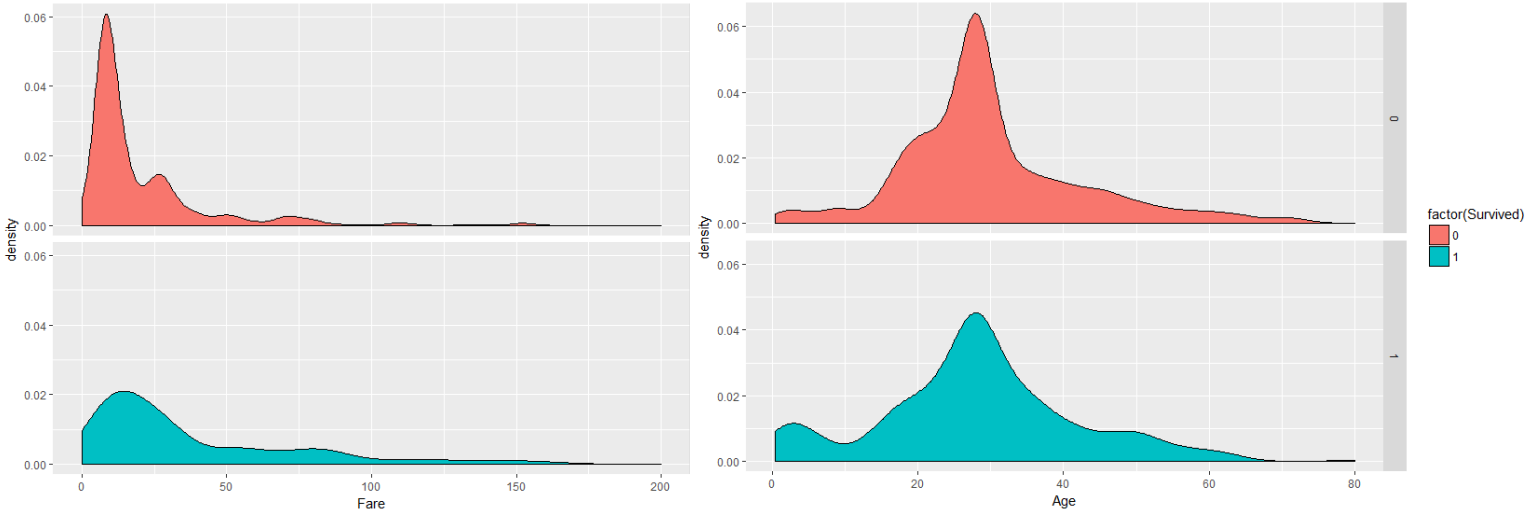
Algunos gráficos explicativos que estudian la distribución de **Survived** frente a variables **categóricas**.



Sobrevivieron más del triple de mujeres que hombres (75% **female** frente a menos del 25% **male**).
Sobrevivieron más del 60% de los pasajeros de primera (**Pclass = 1**), en torno al 50% de los de segunda (**Pclass = 2**), y el 25% de los de tercera (**Pclass = 3**).

Regresión Logística: Ejemplo

Algunos gráficos explicativos que estudian la distribución de **Survived** frente a variables **numéricas**.



No se observan demasiadas diferencias en la distribuciones de **Fare** para los dos valores de **Survived** (salvo un pico pronunciado a tarifas bajas).
Respecto a la distribución de valores de **Age**, las diferencias en también son pequeñas cuando **Survived** $= 0, 1$.

Regresión Logística: Ejemplo

Probamos un modelo en el que describimos la probabilidad de supervivencia según las variables **Sex, Pclass, Age,** y **Fare.**

```
first_model <- glm(Survived ~ Sex + Pclass + Fare + Age,
                   data = titanic,
                   family = binomial(link = 'logit'))

summary(first_model)
```

```
> summary(first_model)

Call:
glm(formula = Survived ~ Sex + Pclass + Fare + Age, family = binomial(link = "logit"),
    data = titanic)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6595  -0.6649  -0.4279   0.6349   2.4318

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.4577090  0.4159040   8.314  < 2e-16 ***
Sexmale     -2.6047131  0.1875578 -13.888  < 2e-16 ***
Pclass2     -1.0706011  0.2854780  -3.750 0.000177 ***
Pclass3     -2.2825075  0.2804322  -8.139 3.98e-16 ***
Fare         0.0007582  0.0020994   0.361 0.717995
Age         -0.0328894  0.0074292  -4.427 9.55e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

La variable **Fare** no parece ser estadísticamente significativa. Podríamos no utilizarla en un segundo modelo.

La instrucción para ajustar un modelo de regresión logística en R es:

```
glm(formula, data, family = binomial(link = 'logit'), ...)
```

Donde

- **`formula`** es una descripción simbólica del modelo que se pretende ajustar.
- **`data`** es un *data frame* que contiene los datos con los que se pretende ajustar el modelo.
- **`family = binomial(link = 'logit')`** refleja que el modelo lineal generalizado es de tipo regresión logística (problema binomial con función link logit)

Regresión Logística: Ejemplo

La salida del modelo son probabilidades, más precisamente $\text{logit}(p)$, definido anteriormente.

Una manera de convertir las predicciones de probabilidad en predicciones de la variable objetivo (**Survived**) es establecer un corte, que puede ser 0.5. De manera que probabilidades de supervivencia mayores del 50% indican que ha sobrevivido.

```
titanic$survived_prediction <- titanic$predictions > 0.5
```

Una vez hecho esto la manera de comprobar si el modelo es bueno es construir una matriz de confusión en la que se comparan las predicciones con los valores reales de la variable objetivo.

```
# Confusion matrix
table(titanic$Survived, titanic$survived_prediction)
```

```
> table(titanic$survived, titanic$survived_prediction)

     FALSE TRUE
  0   472   77
  1   100  242
```

Predicciones

Valores reales

El modelo acierta la supervivencia un 70% de las veces (aprox. 242/(242+100)).
El modelo acierta la no supervivencia un 86% de las veces (aprox. 472/(472+77)).

*Trabaja el Ejemplo **14_logistic_regression.R***

¡No pierdas la pista a los ejemplos en la carpeta correspondiente!

3 Cierre

- Hasta aquí se han visto algunas técnicas básicas de aprendizaje no supervisado y supervisado.

- De hecho, las que se han expuesto se relacionan más con las técnicas estadísticas clásicas pero existen multitud de técnicas del mundo del Machine Learning, como son los Árboles de Decisión, Random Forest, Redes Neuronales y Deep Learning, entre otras.

- Profundizar en el lenguaje R, Python, en técnicas de análisis avanzadas, además de metodologías más complejas y tecnologías Big Data son el paso hacia la formación completa de un Data Scientist.

Dónde encontrar esto y más

An Introduction to Statistical Learning with Applications in R:
http://www-bcf.usc.edu/~gareth/ISL/