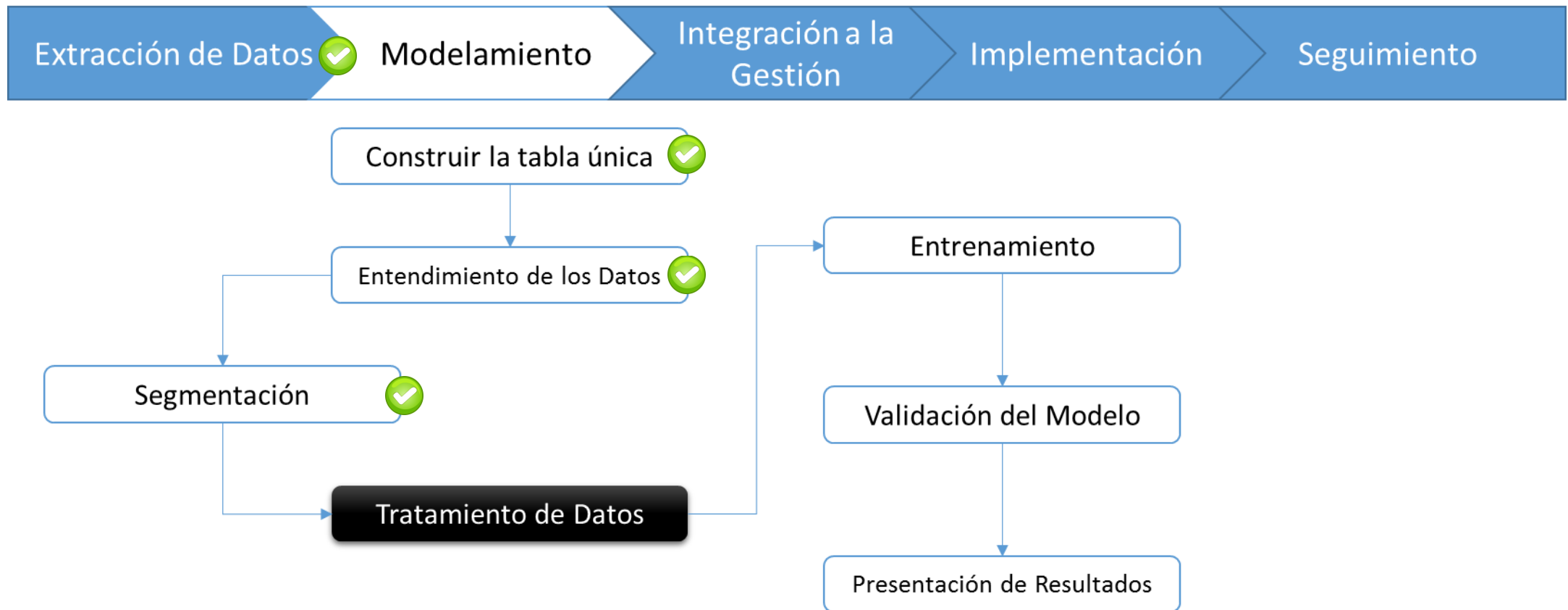


Data Analytics

Tratamiento de Datos

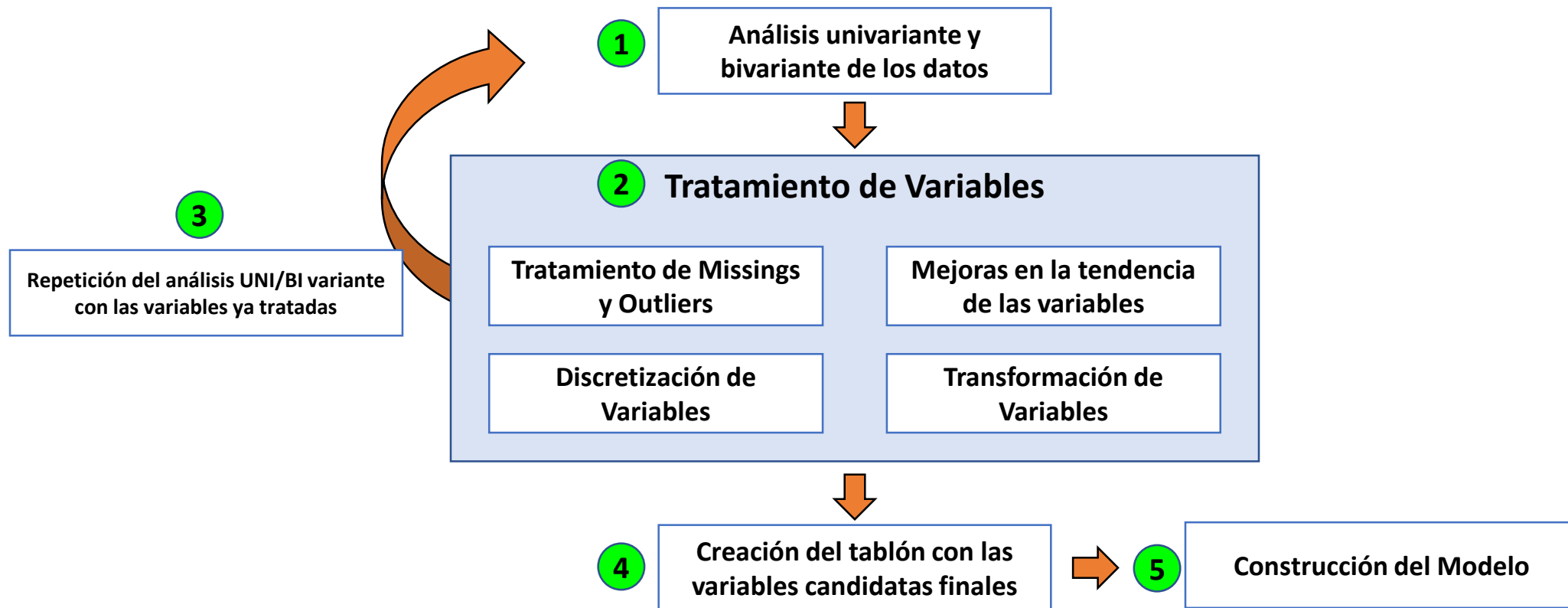
Preprocessing

■ Estamos Aquí:



Introducción

A partir del análisis univariante y bivalente de la extracción se lleva a cabo un tratamiento de variables con el objetivo de perfilar los datos que finalmente serán utilizados en la construcción del modelo.





Tratamiento de Datos

Derivado de los Análisis Univariantes y Bivariantes

Introducción

El tratamiento de variables tiene como objetivo la mejora de la calidad de los datos que serán utilizados en la construcción de modelos, utilizando métodos de tratamiento de missings y outliers, de mejora de la tendencia de variables, discretización y transformación de variables.

- **Análisis y tratamiento de missings** (valores no informados) **y outliers** (atípicos, valores fuera de dominio).

Algunas técnicas son:

- Eliminación de registros
- Sustitución por otro valor
- Inclusión de dummies
- Discretización de la variables

- **Creación de nuevas variables a partir de otras existentes en la tabla única.**

Algunos ejemplos:

- Scalamiento
- Transformar fechas en antigüedad
- Crear ratios, promedios, conteos, máximos, mínimos, etc.



- **Introducir mejoras en la tendencia de las variables.**

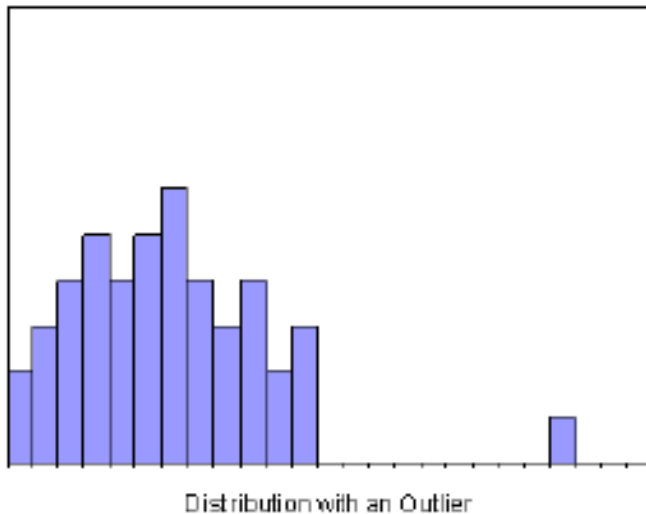
- Tratamiento del Target
- Suavización de variables
- Discretización

- **Creación de nuevas variables a partir de variables continuas, mediante la creación de grupos homogéneos.**

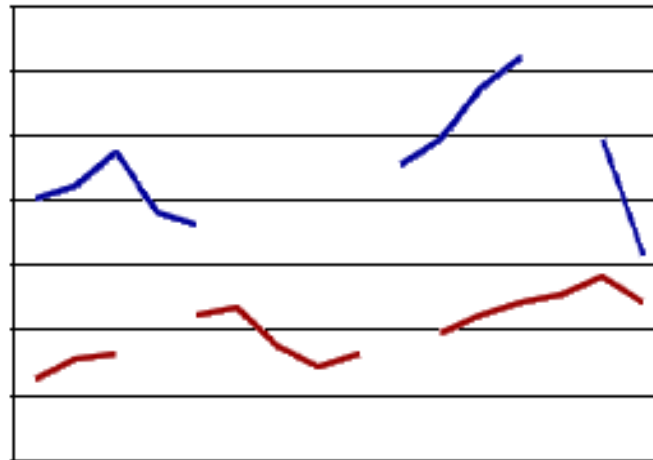
- Creación de grupos de forma experta a partir de análisis de frecuencias o histogramas
- Métodos de clustering
- Discretización automática buscando heterogeneidad del target.

Tratamiento de Missings y Outliers

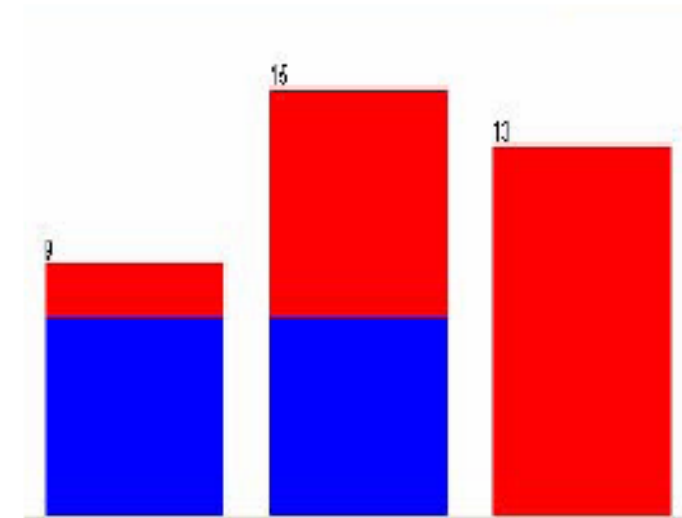
Este tipo de tratamiento mejora la calidad, distribución y tendencia de las variables. Se determinan a partir de los resultados de los análisis Univariantes y Bivariantes.



Valores extremos



Valores null



inconsistentes

Tratamiento de Missings

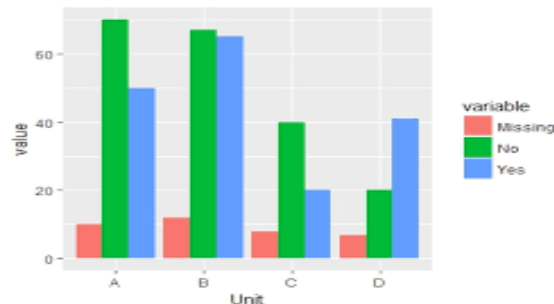
Antes del tratamiento de missings es preciso realizar un análisis experto de los motivos de la desinformación, esto permite determinar cual es la metodología de tratamiento mas idónea.

Identificación:

■ Datos Discretos:

```
Frequencies
Type
Data frame: data
Type: Factor (unordered)
```

	Freq	% Valid	% Valid Cum.	% Total	% Total Cum.
E	19	31.67	31.67	29.69	29.69
B	15	25.00	56.67	23.44	53.12
C	11	18.33	75.00	17.19	70.31
D	8	13.33	88.33	12.50	82.81
A	7	11.67	100.00	10.94	93.75
<NA>	4			6.25	100.00
Total	64	100.00	100.00	100.00	100.00



■ Datos Continuos:

Ozone	Solar.R	Wind	Temp
Min. : 1.00	Min. : 7.0	Min. : 1.700	Min. :57.00
1st Qu.: 18.00	1st Qu.:115.8	1st Qu.: 7.400	1st Qu.:73.00
Median : 31.50	Median :205.0	Median : 9.700	Median :79.00
Mean : 42.13	Mean :185.9	Mean : 9.806	Mean :78.28
3rd Qu.: 63.25	3rd Qu.:258.8	3rd Qu.:11.500	3rd Qu.:85.00
Max. :168.00	Max. :334.0	Max. :20.700	Max. :97.00
NA's :37	NA's :7	NA's :7	NA's :5

Tratamiento de missings

Si el porcentaje de missings de una variable es relevante y se quiere mantener dicha variable en el modelo, es recomendable realizar un tratamiento para no perder un porcentaje elevado de la población. Es preciso establecer un umbral de missings aceptable.

- Reemplazar (Recomendable)

- Por una constante global para reconocimiento del algoritmo
- Por la media del resto de observaciones (de la misma clase)
- Por el valor mas probable obtenido por una técnica de inferencia (Bayes, Arboles, etc.)

- No recomendable

- Ignorar, Existen algoritmos robustos
- Eliminar la variable
- Filtrar las filas

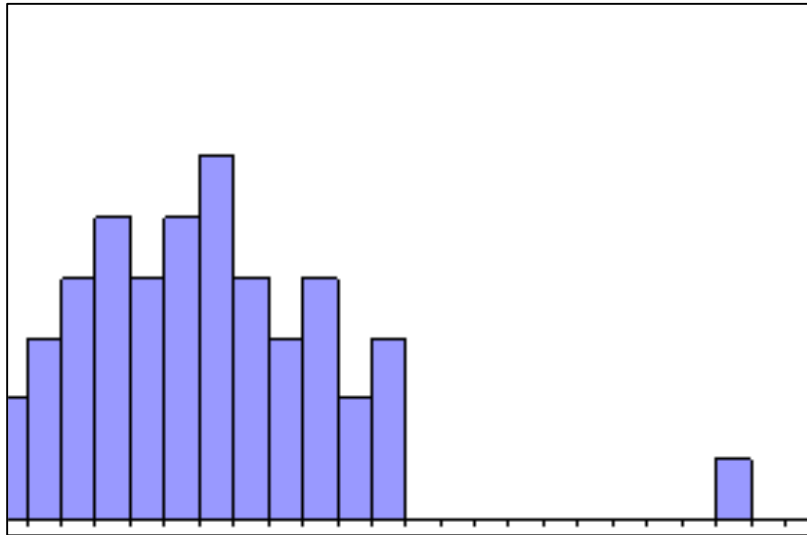
A la hora de realizar el tratamiento es importante:

1. No modificar la distribución de los datos y
2. No cambiar la relación de la variable tratada con el resto de variables

Datos Extremos - Identificación

Método Visual

- Diagrama de Frecuencias
- Diagrama de Cajas (Boxplot)



Método Analítico

- Identificar puntos que se encuentra fuera de la $\mu \pm 3\sigma$
- Las que están fuera del Rango intercuartílico :

$$[Q1 - 1.5 * IRQ, Q3 + 1.5 * IRQ]$$

Datos Extremos -Tratamiento de Cotas

Recomendable

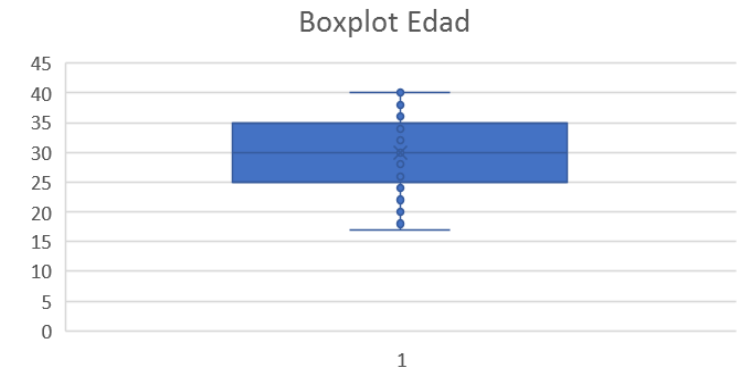
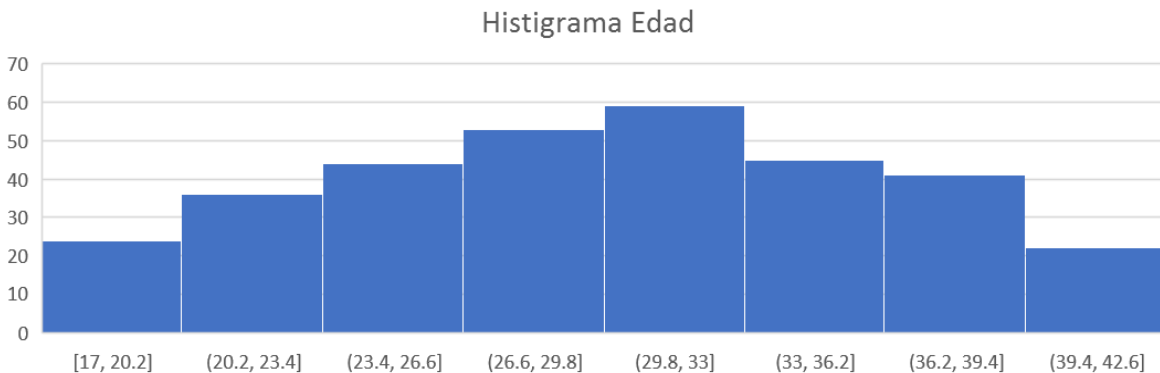
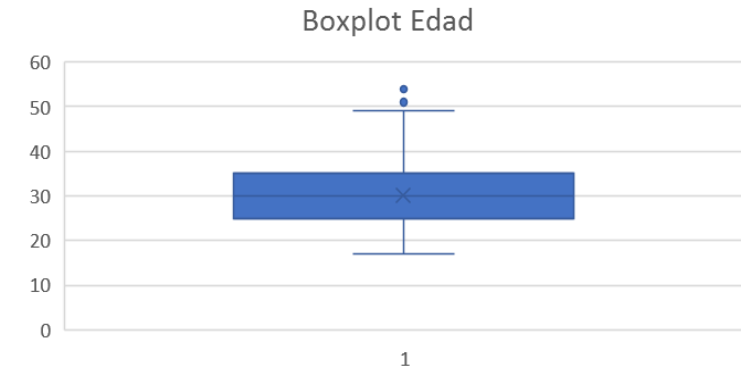
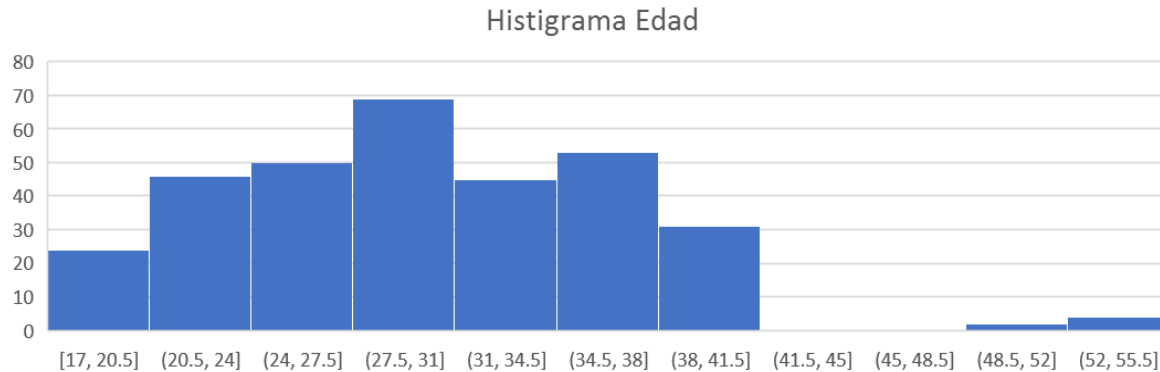
- Discretizar. Transformar un valor continuo en discreto (Muy alto, ..., muy Bajo)
- Imputar Percentiles extremos (Pe. P1 y P99)

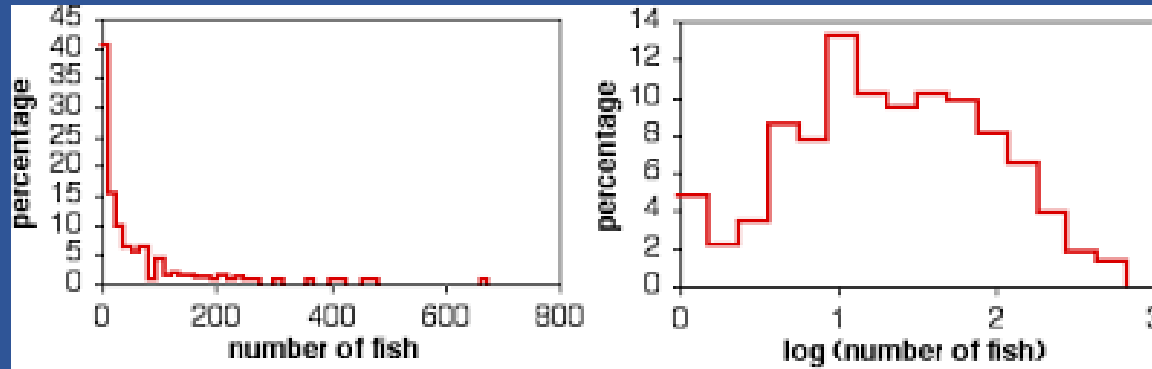
No recomendable

- Ignorar, Existen algoritmos robustos
- Eliminar la variable
- Filtrar las filas

Tratamiento de Cotas

Un atípico es una observación alejada de la mediana y cuya eliminación de la muestra podría suponer un cambio significativo en la estimación del modelo. Su tratamiento no debería distorsionar la distribución inicial de la población.



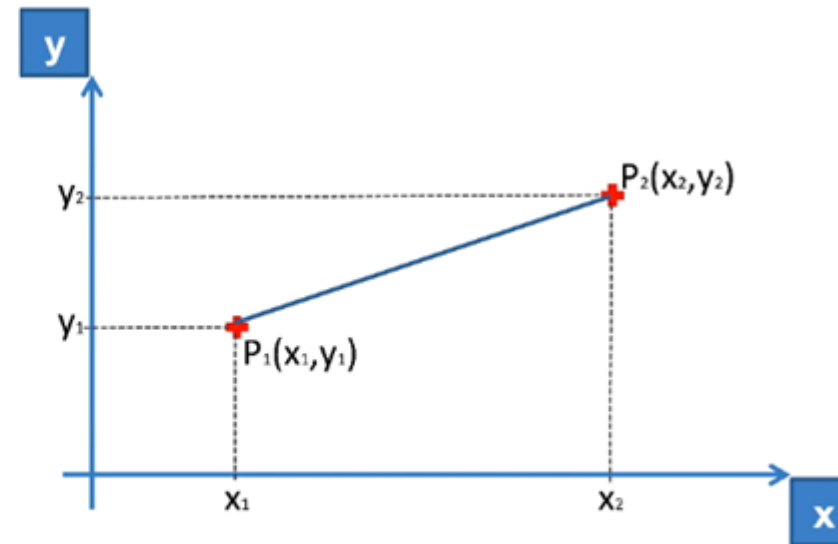


Transformación de Datos

Feature Scaling

- La mayoría de algoritmos de aprendizaje automático, se basa en lo que se llama la distancia euclidiana. Tener valores en diferente escala causará problemas en este calculo.

1	Country	Age	Salary	Purchased
2	France	44	72000	No
3	Spain	27	48000	Yes
4	Germany	30	54000	No
5	Spain	38	61000	No
6	Germany	40	63777.77778	Yes
7	France	35	58000	Yes
8	Spain	38.77777778	52000	No
9	France	48	79000	Yes
10	Germany	50	83000	No
11	France	37	67000	Yes



$$\text{Euclidean Distance between } P_1 \text{ and } P_2 = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Feature Scaling

- Existen varias formas de escalar los datos
- Las mas comunes son las siguientes:

Standardisation	Normalisation
$x_{\text{stand}} = \frac{x - \text{mean}(x)}{\text{standard deviation}(x)}$	$x_{\text{norm}} = \frac{x - \min(x)}{\max(x) - \min(x)}$

- Con esto se esta colocando las variables en el mismo rango en la misma escala, de modo que ninguna variable esté dominada por otra.


Feature Scaling

- Transformando las variables de valores grandes y muy diferentes a valores pequeños y similares.
- Incluso si los algoritmos no están basados en la distancia euclidiana, es conveniente hacer el escalamiento para que el algoritmo converja mucho mas rápido (Arboles de decisión).
- El escalamiento de variables categóricas no es recomendable dado que perdemos interpretación del modelo resultante.
- Considerar lo siguiente, respeto al escalamiento de la variable target:
 - Modelos de regresión: recomendable
 - Modelos de clasificación: no escalar la variable target

Feature Scaling

■ Hands On en R:

```
# Feature Scaling  
dataset[,2:3] = scale(dataset[,2:3])
```



	Country	Age	Salary	Purchased
1	1	44.00000	72000.00	0
2	2	27.00000	48000.00	1
3	3	30.00000	54000.00	0
4	2	38.00000	61000.00	0
5	3	40.00000	63777.78	1
6	1	35.00000	58000.00	1
7	2	38.77778	52000.00	0
8	1	48.00000	79000.00	1
9	3	50.00000	83000.00	0
10	1	37.00000	67000.00	1



	Country	Age	Salary	Purchased
1	1	0.7199314	0.7110128	0
2	2	-1.6236751	-1.3643758	1
3	3	-1.2100975	-0.8455287	0
4	2	-0.1072238	-0.2402070	0
5	3	0.1684946	0.0000000	1
6	1	-0.5208015	-0.4996306	1
7	2	0.0000000	-1.0184777	0
8	1	1.2713683	1.3163344	1
9	3	1.5470867	1.6622325	0
10	1	-0.2450830	0.2786401	1

Feature Scaling

- Una forma muy común de reducir la escala de una variable es utilizando el logaritmo. A este procedimiento también se le conoce como suavización de datos. Se aplica cuando se tienen valores numéricos muy grandes y de alta variabilidad

Logaritmo natural, en base 2 o en base 10.

$$v' = \log_{10}(v)$$

Muestreo Training y Test

Training and test Set

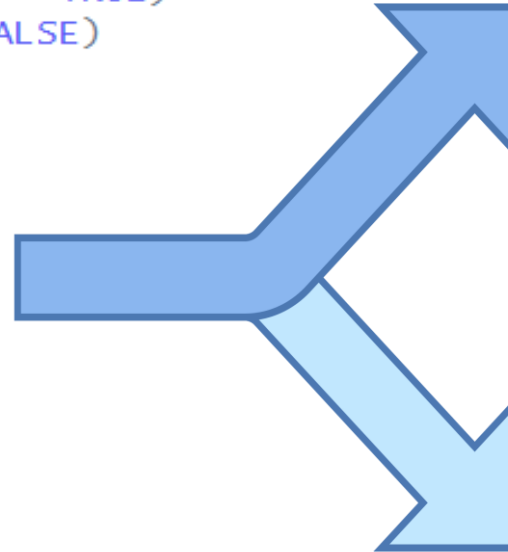
- Dividir la data en dos muestras, es una técnica muy utilizada para evaluar los resultados de un análisis estadístico y garantizar que son independientes de las dos muestras (cross-validation)
- La muestra de train, será con la cual se realiza el entrenamiento y como consecuencia se obtendrá una predicción (formula o regla lógica)
- La muestra de test, permitirá evaluar la predicción. Aplicando la regla obtenida, a este nuevo set de datos (target real vs target estimado)
- Solo aplica para aprendizaje supervisado.
- El muestreo tiene que estar balanceado por la variable target.
- La división de las observaciones recomendada es:
 - training 60-80% del total
 - test 40-20% del total
- El porcentaje de división dependerá del algoritmo seleccionado.

Training and test Set

■ Hands On en R:

```
# install.packages('caTools')
library(caTools)
set.seed(123)
split = sample.split(dataset$Purchased, SplitRatio = 0.8)
training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)
```

	Country	Age	Salary	Purchased
1	1	44.00000	72000.00	0
2	2	27.00000	48000.00	1
3	3	30.00000	54000.00	0
4	2	38.00000	61000.00	0
5	3	40.00000	63777.78	1
6	1	35.00000	58000.00	1
7	2	38.77778	52000.00	0
8	1	48.00000	79000.00	1
9	3	50.00000	83000.00	0
10	1	37.00000	67000.00	1



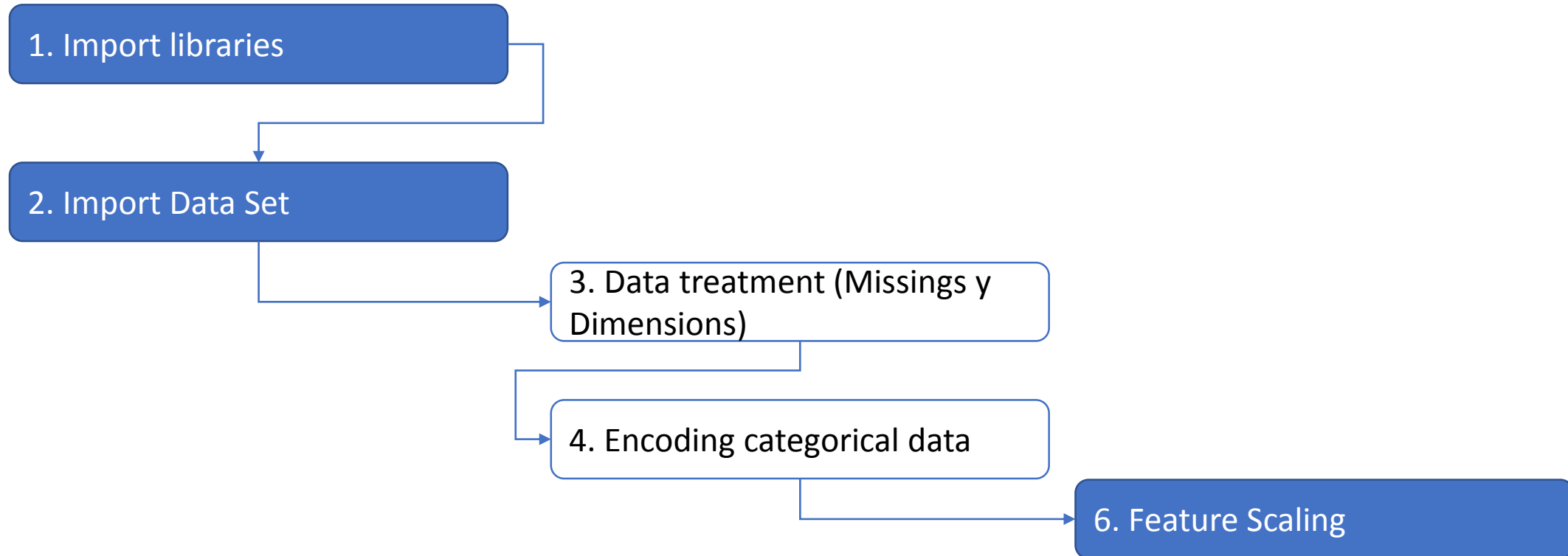
	Country	Age	Salary	Purchased
1	1	44.00000	72000.00	0
2	2	27.00000	48000.00	1
3	3	30.00000	54000.00	0
4	2	38.00000	61000.00	0
5	3	40.00000	63777.78	1
7	2	38.77778	52000.00	0
8	1	48.00000	79000.00	1
10	1	37.00000	67000.00	1

	Country	Age	Salary	Purchased
6	1	35	58000	1
9	3	50	83000	0

Data Preprocessing Template

Data Preprocessing Template

■ Workflow:



Data Preprocessing Template

data_preprocessing_template_final.R

```
1 # Data Preprocessing Template
2
3 # Importing the dataset
4 dataset = read.csv('Data.csv')
5
6 # Taking care of missing data
7 dataset$Age = ifelse(is.na(dataset$Age),
8                     ave(dataset$Age, FUN = function(x) mean(x, na.rm = TRUE)),
9                     dataset$Age)
10 dataset$Salary = ifelse(is.na(dataset$Salary),
11                        ave(dataset$Salary, FUN = function(x) mean(x, na.rm = TRUE)),
12                        dataset$Salary)
13
14 # Encoding categorical data
15 dataset$Country = factor(dataset$Country,
16                          levels = c('France', 'Spain', 'Germany'),
17                          labels = c(1, 2, 3))
18 dataset$Purchased = factor(dataset$Purchased,
19                             levels = c('No', 'Yes'),
20                             labels = c(0, 1))
21 # Splitting the dataset into the Training set and Test set
22 # install.packages('caTools')
23 library(caTools)
24 set.seed(123)
25 split = sample.split(dataset$Purchased, SplitRatio = 0.8)
26 training_set = subset(dataset, split == TRUE)
27 test_set = subset(dataset, split == FALSE)
28
29 # Feature Scaling
30 training_set = scale(training_set)
31 test_set = scale(test_set)
```

data_preprocessing_template_final.py

```
1 # Data Preprocessing Template
2
3 # Importing the libraries
4 import numpy as np
5 import matplotlib.pyplot as plt
6 import pandas as pd
7
8 # Importing the dataset
9 dataset = pd.read_csv('Data.csv')
10 X = dataset.iloc[:, :-1].values
11 y = dataset.iloc[:, 3].values
12
13 # Taking care of missing data
14 from sklearn.preprocessing import Imputer
15 imputer = Imputer(missing_values = 'NaN', strategy = 'mean', axis = 0)
16 imputer = imputer.fit(X[:, 1:3])
17 X[:, 1:3] = imputer.transform(X[:, 1:3])
18
19 # Encoding categorical data
20 # Encoding the Independent Variable
21 from sklearn.preprocessing import LabelEncoder, OneHotEncoder
22 labelencoder_X = LabelEncoder()
23 X[:, 0] = labelencoder_X.fit_transform(X[:, 0])
24 onehotencoder = OneHotEncoder(categorical_features = [0])
25 X = onehotencoder.fit_transform(X).toarray()
26 # Encoding the Dependent Variable
27 labelencoder_y = LabelEncoder()
28 y = labelencoder_y.fit_transform(y)
29
30 # Splitting the dataset into the Training set and Test set
31 from sklearn.cross_validation import train_test_split
32 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
33
34 # Feature Scaling
35 from sklearn.preprocessing import StandardScaler
36 sc_X = StandardScaler()
37 X_train = sc_X.fit_transform(X_train)
38 X_test = sc_X.transform(X_test)
39 sc_y = StandardScaler()
40 y_train = sc_y.fit_transform(y_train)
```