KIBERNUM
IT Academy

Módulo 2
Clase 5

Obtención y Preparación de Datos

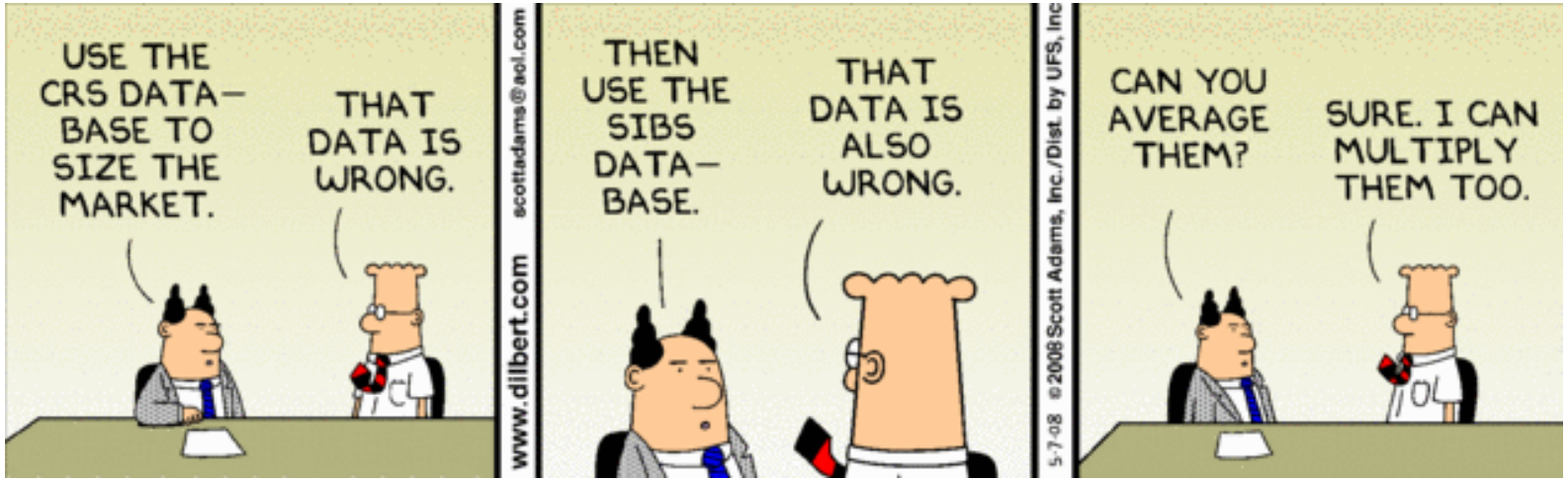
Limpieza de Datos

- Calidad de datos
- Valores perdidos
- Técnicas de imputación
- Valores atípicos

KIBERNUM

Calidad de Datos

KIBERNUM


USE THE CRS DATA-BASE TO SIZE THE MARKET.
THAT DATA IS WRONG.
www.dilbert.com scottadams@aol.com
THEN USE THE SIBS DATA-BASE.
THAT DATA IS ALSO WRONG.
5-7-08 ©2008 Scott Adams, Inc./Dist. by UFS, Inc
CAN YOU AVERAGE THEM?
SURE. I CAN MULTIPLY THEM TOO.

KIBERNUM


BREAKING BAD
DATA
• 23% of customer data is believed to be inaccurate.
• 92% of organizations have experienced problems as a result of inaccurate data.
WHAT CAN YOUR BUSINESS DO ABOUT IT?
QA2L.COM

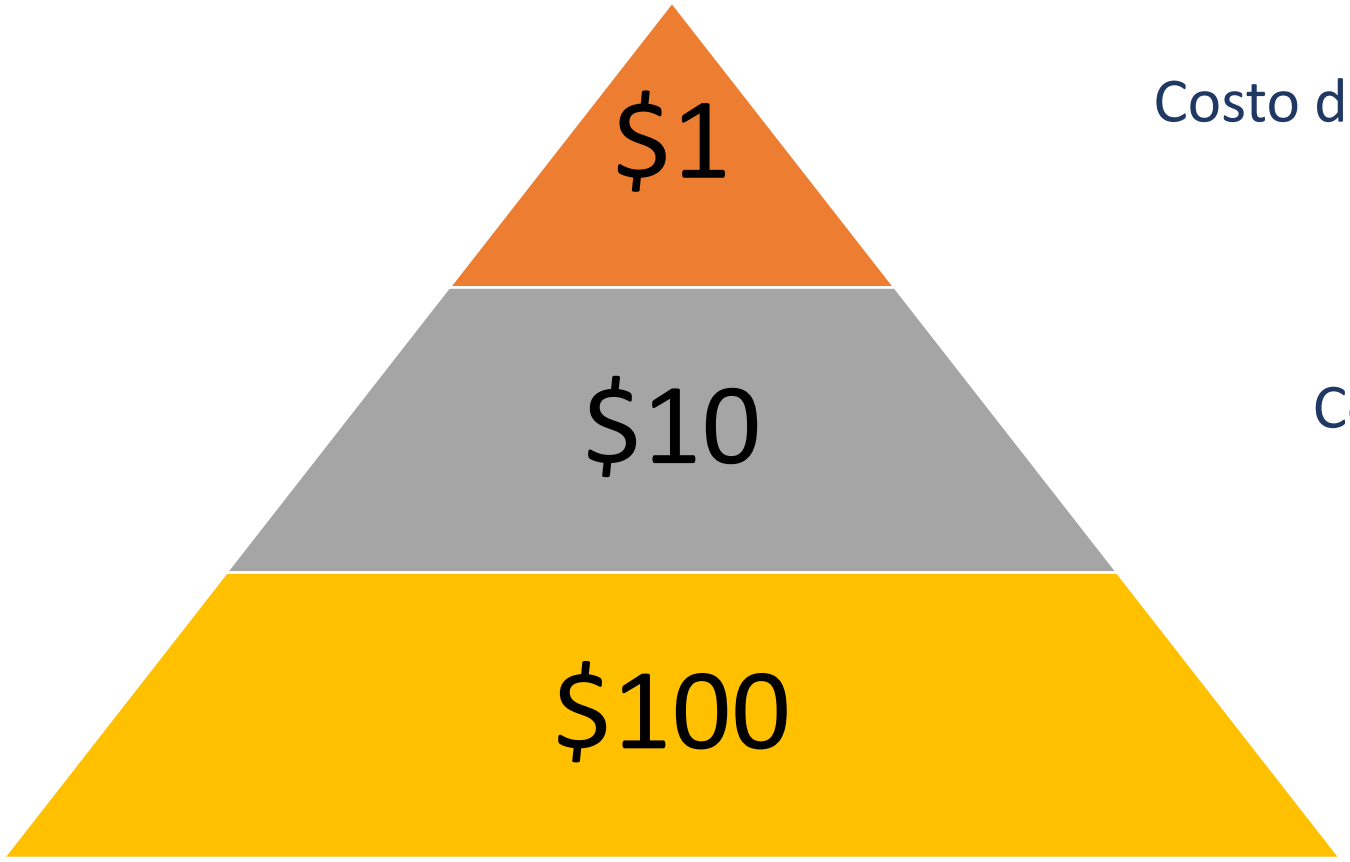
KIBERNUM

Bad Data = Bad DECISIONS

Cuando los datos son incorrectos, pueden inducir a conclusiones falsas y a tomar decisiones incorrectas.

KIBERNUM

La regla del 1-10-100



Costo de Prevención

Costo de Corrección

Costo de Falla

La regla del 1-10-100 es un concepto de gestión de la calidad desarrollado por G. Loabovitz y por Y. Chang que es usado para cuantificar los costos ocultos de la baja calidad.

Calidad de Datos

Si el valor almacenado es el valor correcto

Relacionado con los datos duplicados

Que se mantenga inalterada ante accidentes o eventos

Cociente de valores omitidos sobre el número de valores totales

Corrección de contradicciones y anomalías sintácticas

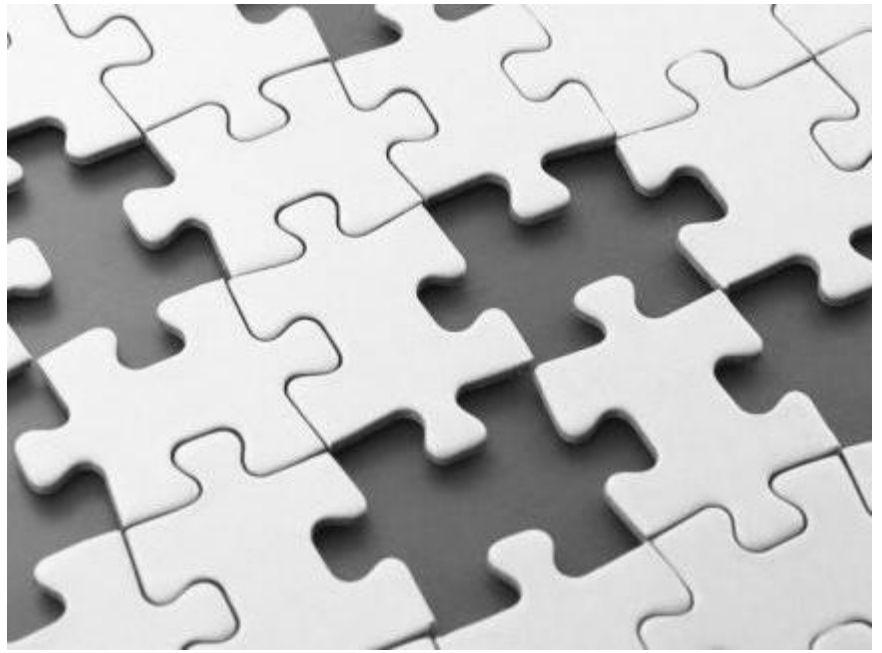
Irregularidades

Limpieza de Datos

Causas de los Valores Perdidos

Algunos casos típicos, por la cual se generan valores perdidos en los datos son los siguientes:

- El usuario olvidó completar un campo de información.
- Se perdieron datos durante el proceso manual de transferencia desde una fuente de datos legada.
- Hubo un error de programación.
- El usuario simplemente no quiso compartir cierta información.



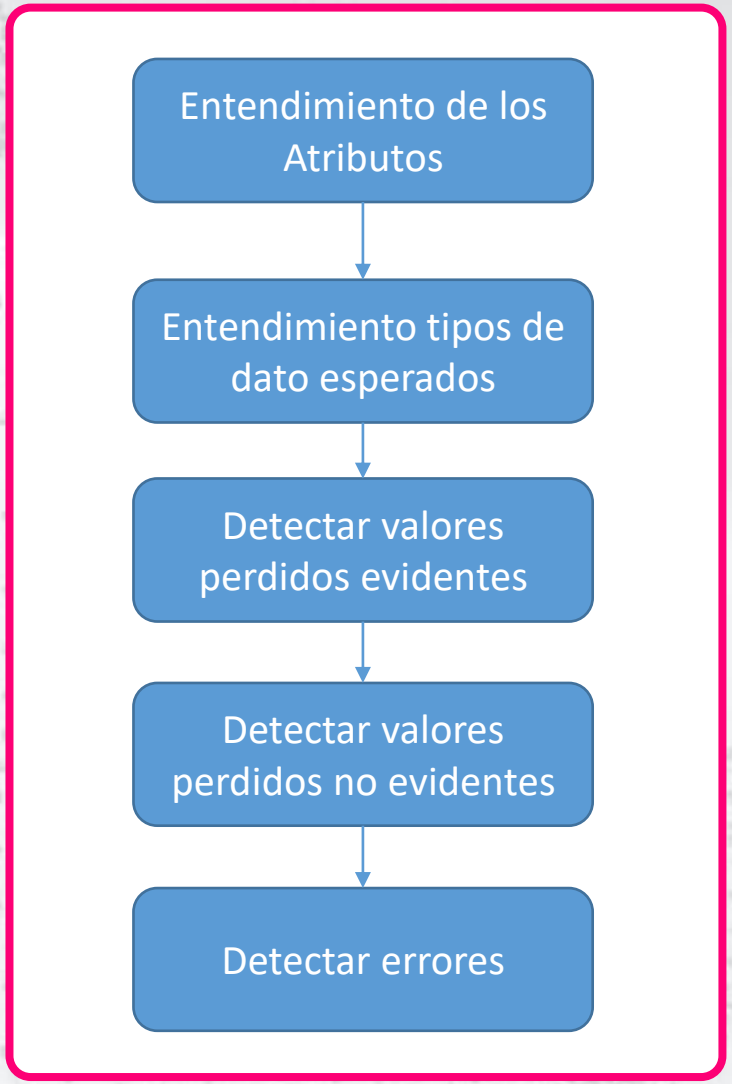
KIBERNUM

Plan de Acción

Causas de los Valores Perdidos

Lo primero, es entender los datos que estamos trabajando (Análisis Exploratorio):

- ¿Qué atributos (features) posee?
- ¿Cuáles son los tipos de dato que esperamos de cada uno de ellos?
- ¿Hay valores perdidos que son obvios de encontrar? (con una librería o software).
- ¿Hay otro tipo de valores perdidos que no son tan obvios de encontrar? (no son tan fáciles de detectar con una librería o software).
- Valores erróneos.



Dataset de Ejemplo

Sea el siguiente **dataset** que corresponde a una empresa dedicada al corretaje de propiedades. Cada registro corresponde a una propiedad de su inventario.

Nótese que se aprecian datos faltantes y erróneos.

```
PID,ST_NUM,ST_NAME,OWN_OCCUPIED,NUM_BEDROOMS,NUM_BATH,SQ_FT
100001000,104,PUTNAM,Y,3,1,1000
100002000,197,LEXINGTON,N,3,1.5,--
100003000,,LEXINGTON,N,n/a,1,850
100004000,201,BERKELEY,12,1,NaN,700
,203,BERKELEY,Y,3,2,1600
100006000,207,BERKELEY,Y,NA,1,800
100007000,NA,WASHINGTON,,2,HURLEY,950
100008000,213,TREMONT,Y,1,1,
100009000,215,TREMONT,Y,na,2,1800
```

Analizando los atributos y sus tipos de datos

A continuación se describen las columnas de este **dataset** y los tipos de datos esperados.

Atributo	Descripción	Tipo de Dato Esperado
PID	Identificador de la propiedad	Numérico
ST_NUM	Numeración dirección propiedad	Numérico
ST_NAME	Calle dirección propiedad	Categórica (string)
OWN_OCCUPIED	Si se encuentra ocupada actualmente	Categórica (string Y ó N)
NUM_BEDROOMS	Cantidad de dormitorios	Numérico (entero o decimal)
NUM_BATH	Cantidad de baños	Numérico (entero o decimal)
SQ_FT	Superficie en pies cuadrados	Numérico (entero o decimal)

Identificación de valores perdidos estándar

Los valores perdidos estándar son aquellos que pueden ser identificados fácilmente por un software o por una librería porque son reconocibles.

```
df = pd.read_csv('real-estate.csv')
```

```
df
```

	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	100001000.0	104.0	PUTNAM	Y	3	1	1000
1	100002000.0	197.0	LEXINGTON	N	3	1.5	--
2	100003000.0	NaN	LEXINGTON	N	NaN	1	850
3	100004000.0	201.0	BERKELEY	12	1	NaN	700
4	NaN	203.0	BERKELEY	Y	3	2	1600
5	100006000.0	207.0	BERKELEY	Y	NaN	1	800
6	100007000.0	NaN	WASHINGTON	NaN	2	HURLEY	950
7	100008000.0	213.0	TREMONT	Y	1	1	NaN
8	100009000.0	215.0	TREMONT	Y	na	2	1800

La librería Pandas los representa con **NaN.**

KIBERNUM

Identificación de valores perdidos estándar

La librería Pandas identifica fácilmente los valores perdidos evidentes, el método isnull(), retorna una matriz con True en donde determina que existe un valor perdido.

Con esta instrucción podemos contabilizar la cantidad de valores perdidos evidentes que existe en un dataframe.

```
df.isnull()
```

	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False
2	False	True	False	False	True	False	False
3	False	False	False	False	False	True	False
4	True	False	False	False	False	False	False
5	False	False	False	False	True	False	False
6	False	True	False	True	False	False	False
7	False	False	False	False	False	False	True
8	False	False	False	False	False	False	False

```
df.isnull().sum()
```

```
PID             1
ST_NUM          2
ST_NAME         0
OWN_OCCUPIED    1
NUM_BEDROOMS    2
NUM_BATH        1
SQ_FT           1
dtype: int64
```

Valores perdidos no estándar

	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	100001000.0	104.0	PUTNAM	Y	3	1	1000
1	100002000.0	197.0	LEXINGTON	N	3	1.5	--
2	100003000.0	NaN	LEXINGTON	N	NaN	1	850
3	100004000.0	201.0	BERKELEY	12	1	NaN	700
4	NaN	203.0	BERKELEY	Y	3	2	1600
5	100006000.0	207.0	BERKELEY	Y	NaN	1	800
6	100007000.0	NaN	WASHINGTON	NaN	2	HURLEY	950
7	100008000.0	213.0	TREMONT	Y	1	1	NaN
8	100009000.0	215.0	TREMONT	Y	na	2	1800

```
df.isnull()
```

	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False
2	False	True	False	False	True	False	False
3	False	False	False	False	False	True	False
4	True	False	False	False	False	False	False
5	False	False	False	False	True	False	False
6	False	True	False	True	False	False	False
7	False	False	False	False	False	False	True
8	False	False	False	False	False	False	False

Acá apreciamos que cuando se hizo ingreso de la información, también se utilizó una forma no estándar de señalar valores vacíos (na, --)

KIBERNUM

Corrigiendo valores perdidos no-estándar

Especificamos las etiquetas que deseamos que sean reconocidas como, valores NaN.

```
df = pd.read_csv('real-estate.csv', na_values=['na','--'])
df
```

	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	100001000.0	104.0	PUTNAM	Y	3.0	1	1000.0
1	100002000.0	197.0	LEXINGTON	N	3.0	1.5	NaN
2	100003000.0	NaN	LEXINGTON	N	NaN	1	850.0
3	100004000.0	201.0	BERKELEY	12	1.0	NaN	700.0
4	NaN	203.0	BERKELEY	Y	3.0	2	1600.0
5	100006000.0	207.0	BERKELEY	Y	NaN	1	800.0
6	100007000.0	NaN	WASHINGTON	NaN	2.0	HURLEY	950.0
7	100008000.0	213.0	TREMONT	Y	1.0	1	NaN
8	100009000.0	215.0	TREMONT	Y	NaN	2	1800.0

Valores perdidos no esperados

Hay valores que no son detectados como perdidos, pero son no-esperados. Estos valores deberán ser procesados posteriormente en la etapa de Data Wrangling.

	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	100001000.0	104.0	PUTNAM	Y	3	1	1000
1	100002000.0	197.0	LEXINGTON	N	3	1.5	--
2	100003000.0	NaN	LEXINGTON	N	NaN	1	850
3	100004000.0	201.0	BERKELEY	12	1	NaN	700
4	NaN	203.0	BERKELEY	Y	3	2	1600
5	100006000.0	207.0	BERKELEY	Y	NaN	1	800
6	100007000.0	NaN	WASHINGTON	NaN	2	HURLEY	950
7	100008000.0	213.0	TREMONT	Y	1	1	NaN
8	100009000.0	215.0	TREMONT	Y	na	2	1800

Valores perdidos no esperados

Por lo pronto, acá se presenta una forma sencilla verificar el contenido de una columna para chequear valores no esperados.

```
# valores únicos de una columna
df['OWN_OCCUPIED'].unique()
```

```
array(['Y', 'N', '12', nan], dtype=object)
```

```
# cantidad de valores únicos de la columna
df['OWN_OCCUPIED'].unique()
```

```
array(['Y', 'N', '12', nan], dtype=object)
```

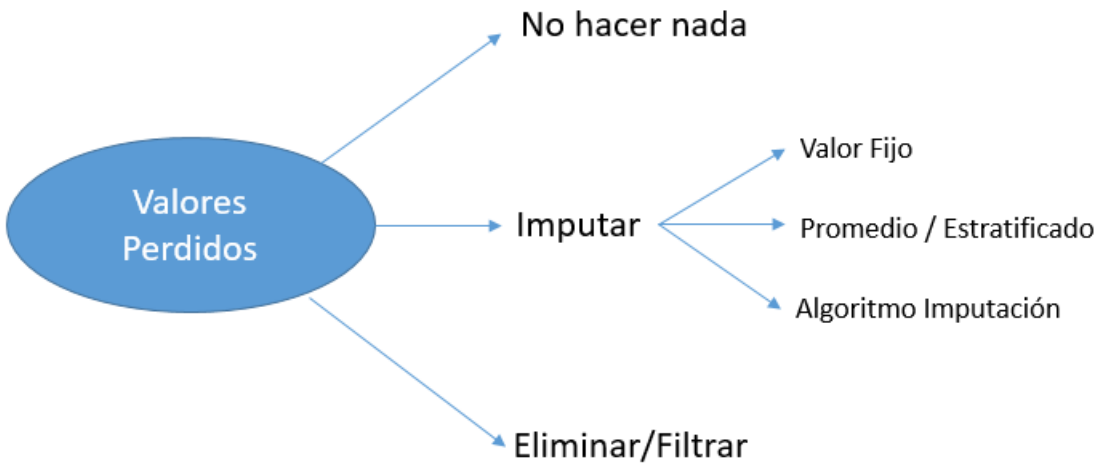
```
# fercuencia de cada valor único en una columna
df['OWN_OCCUPIED'].value_counts()
```

```
Y     5
N     2
12    1
Name: OWN_OCCUPIED, dtype: int64
```

Tratamiento de los valores perdidos

KIBERNUM

Tratamiento de los valores perdidos



Una vez identificados los valores perdidos debemos decidir qué hacer con ellos. La respuesta dependerá del contexto del problema. Las siguientes son posibles alternativas:

KIBERNUM

Filtrado de filas con valores perdidos

En este método, se eliminan las filas que contienen valores perdidos. Sin embargo, este método es recomendado si los valores perdidos son pocos en comparación con todo el set de datos, ya que podría suceder que quede una candidad de registros insuficientes para llevar a cabo los análisis.

```
df.dropna()
```

	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	100001000.0	104.0	PUTNAM	Y	3.0	1	1000.0

El método dropna() por defecto elimina todas las filas que contienen al menos un valor perdido. En este ejemplo, sólo quedó una única fila que no contenía ningún valor perdido.

Filtrado de columnas con valores perdidos

Si la cantidad de valores perdidos en una columna es alto, se podría considerar la opción de realizar una eliminación de la columna.

```
df.dropna(axis=1)
```

	ST_NAME
0	PUTNAM
1	LEXINGTON
2	LEXINGTON
3	BERKELEY
4	BERKELEY
5	BERKELEY
6	WASHINGTON
7	TREMONT
8	TREMONT

El método **dropna()** también, puede eliminar columnas que contengan al menos un valor perdido, especificando el parámetro **axis=1** . En este caso, sólo quedó una sola columna que no contenía valores nulos.

Por defecto, axis=0, es decir, el método elimina filas

Filtrando subset de valores perdidos

Utilizar dropna() "a secas" puede ser un tanto exagerado, a veces tomaremos la decisión de eliminar o no un registro en base a las columnas claves. Para esto, podemos utilizar el parámetro subset.

```
# filtrar subsets con valores perdidos
df.dropna(subset=['SQ_FT','NUM_BEDROOMS'])
```

	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	100001000.0	104.0	PUTNAM	Y	3.0	1	1000.0
3	100004000.0	201.0	BERKELEY	12	1.0	NaN	700.0
4	NaN	203.0	BERKELEY	Y	3.0	2	1600.0
6	100007000.0	NaN	WASHINGTON	NaN	2.0	HURLEY	950.0

En este caso hemos especificado que botaremos las filas con valores nulos en las columnas SQ_FT y NUM_BEDROOMS.

Filtrando subset de valores perdidos

Con el parámetro thresh se puede setear la cantidad mínima de valores no nulos aceptables para que una columna no sea borrada.

```
# umbral para filtrar valores perdidos
df.dropna(thresh=6)
```

	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	100001000.0	104.0	PUTNAM	Y	3.0	1	1000.0
1	100002000.0	197.0	LEXINGTON	N	3.0	1.5	NaN
3	100004000.0	201.0	BERKELEY	12	1.0	NaN	700.0
4	NaN	203.0	BERKELEY	Y	3.0	2	1600.0
5	100006000.0	207.0	BERKELEY	Y	NaN	1	800.0
7	100008000.0	213.0	TREMONT	Y	1.0	1	NaN
8	100009000.0	215.0	TREMONT	Y	NaN	2	1800.0

En este caso, se ha definido que las filas deben tener al menos 6 valores no nulos, caso contrario son filtrados. Nótese las filas que desaparecieron en el set de datos.

Imputación de un valor fijo

La forma más sencilla de realizar una imputación, es mediante la imputación de un valor fijo. En este ejemplo, se han reemplazado los valore nulos por un valor fijo igual a cero. Esta no siempre es la mejor opción, debe estudiarse caso a caso puesto que se está asumiendo un valor ficticio.

```
# imputación valor fijo
df.fillna(value=0)
```

	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	100001000.0	104.0	PUTNAM	Y	3.0	1	1000.0
1	100002000.0	197.0	LEXINGTON	N	3.0	1.5	0.0
2	100003000.0	0.0	LEXINGTON	N	0.0	1	850.0
3	100004000.0	201.0	BERKELEY	12	1.0	0	700.0
4	0.0	203.0	BERKELEY	Y	3.0	2	1600.0
5	100006000.0	207.0	BERKELEY	Y	0.0	1	800.0
6	100007000.0	0.0	WASHINGTON	0	2.0	HURLEY	950.0
7	100008000.0	213.0	TREMONT	Y	1.0	1	0.0
8	100009000.0	215.0	TREMONT	Y	0.0	2	1800.0

Imputación del valor promedio, moda o mediana

El método de imputación de un promedio, moda o mediana, entrega un valor más razonable para una columna, al menos más razonable que imputar el valor cero en la lámina anterior. Nótese que en este caso el valor resultante imputado fue de 1100 ft.

```
# imputar valor promedio en una columna numérica
df['SQ_FT'] = df['SQ_FT'].fillna(df['SQ_FT'].mean())
df
```

	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	100001000.0	104.0	PUTNAM	Y	3.0	1	1000.0
1	100002000.0	197.0	LEXINGTON	N	3.0	1.5	1100.0
2	100003000.0	NaN	LEXINGTON	N	NaN	1	850.0
3	100004000.0	201.0	BERKELEY	12	1.0	NaN	700.0
4	NaN	203.0	BERKELEY	Y	3.0	2	1600.0
5	100006000.0	207.0	BERKELEY	Y	NaN	1	800.0
6	100007000.0	NaN	WASHINGTON	NaN	2.0	HURLEY	950.0
7	100008000.0	213.0	TREMONT	Y	1.0	1	1100.0
8	100009000.0	215.0	TREMONT	Y	NaN	2	1800.0

Otros métodos avanzados de imputación

Otras formas más avanzadas de realizar imputaciones son las siguientes:

- **Imputación estratificada:** agrupamos los registros en base a un criterio o estrato y en base al promedio, mediana o moda del estrato realizamos la imputación.
- **Modelos regresivos:** en el caso de variables numéricas, como por ejemplo, el valor de SQ_FT, se podría elaborar una regresión lineal en base a los atributos NUM_BEDROOMS y NUM_BATH.
- **Modelos de aprendizaje:** A veces se emplean métodos de aprendizaje de máquina tales como K-Means para estimar y reemplazar la data perdida. Este método podría ser utilizado tanto para determinar variables numéricas como categóricas.

Valores Atípicos

KIBERNUM

¿Qué es un valor atípico?

Un valor atípico, en un set de datos, corresponde a una observación que es numéricamente distantes del resto de los datos, extremadamente grande o extremadamente pequeña. Un valor atípico puede generar un efecto desproporcionado en los resultados estadísticos, como la media, lo cual puede conducir a interpretaciones engañosas.

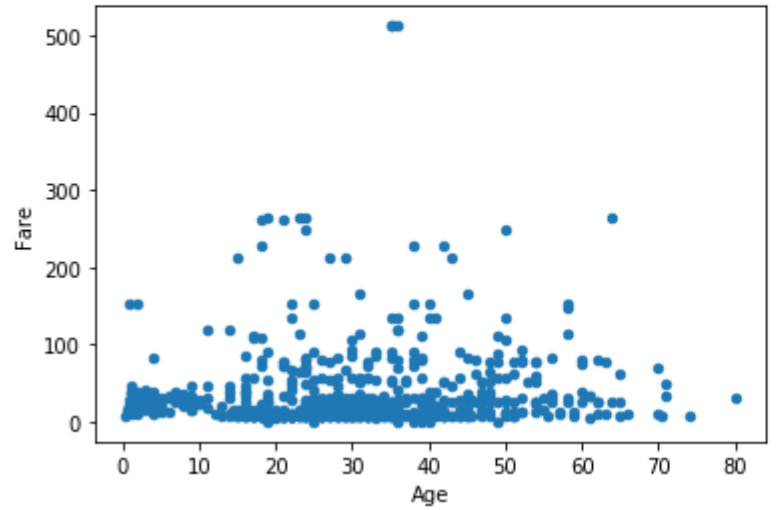
Un valor atípico, llamado a veces anomalía, podría deberse a un fenómeno único (por ejemplo, una lista de clientes de un banco con una persona de 124 años) o bien podría ser producido por un error (por ejemplo, ingresaron 124 en vez de 24 años al momento de crear al cliente).

Sea cual sea el dato, es conveniente identificar la presencia de valores atípicos o anomalías en los datos.

```
df = pd.read_csv('titanic.csv')
```

```
df.plot(kind='scatter', x='Age', y='Fare')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x211d3e4ad08>
```



KIBERNUM

¿Qué es un valor atípico?

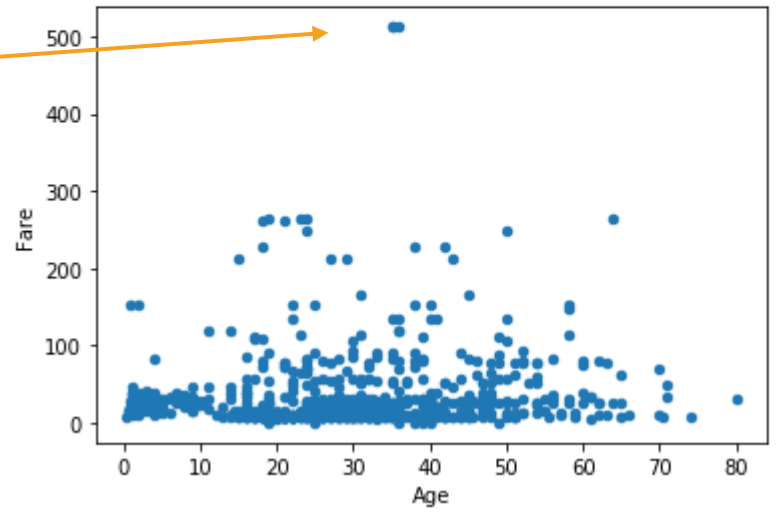
Nótese en el siguiente diagrama de dispersión la presencia de un valor extremo por sobre las 500 libras en el valor del pasaje pagado en el dataset del Titanic.

Nótese también, cómo la media de los valores tiende a alejarse de la mediana. Esto ocurre porque la media corresponde a promedio aritmético, el cual se ve distorsionado por causa del valor extremadamente alto.

```
df = pd.read_csv('titanic.csv')
```

```
df.plot(kind='scatter', x='Age', y='Fare')
```

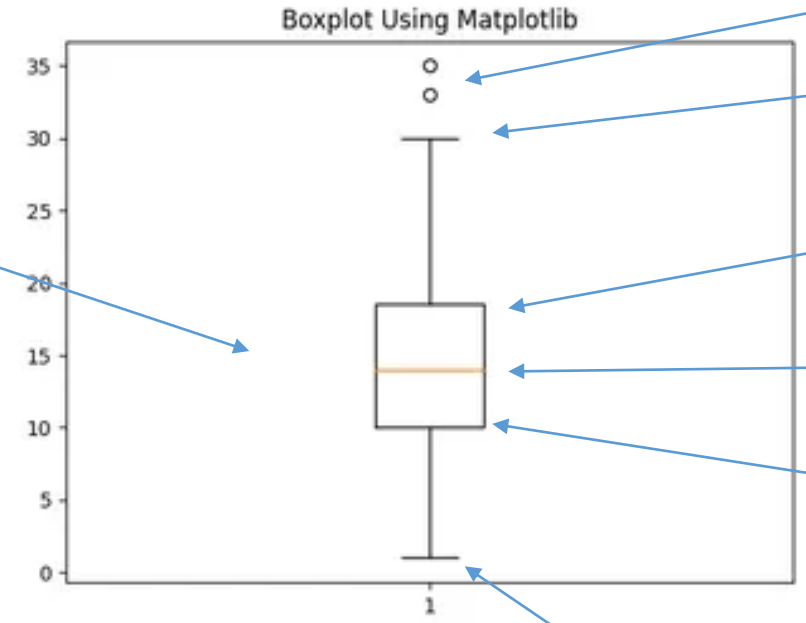
```
<matplotlib.axes._subplots.AxesSubplot at 0x211d3e4ad08>
```




```
df['Fare'].describe()
```

```
count    891.000000
mean      32.204208
std       49.693429
min        0.000000
25%        7.910400
50%       14.454200
75%       31.000000
max      512.329200
Name: Fare, dtype: float64
```

Qué es un valor atípico



Valores atípicos

Valor máximo

Rango intercuartil

Tercer Cuartil (75% de la muestra)

Mediana (50% de la muestra)

Primer Cuartil (25% de la muestra)

Valor mínimo

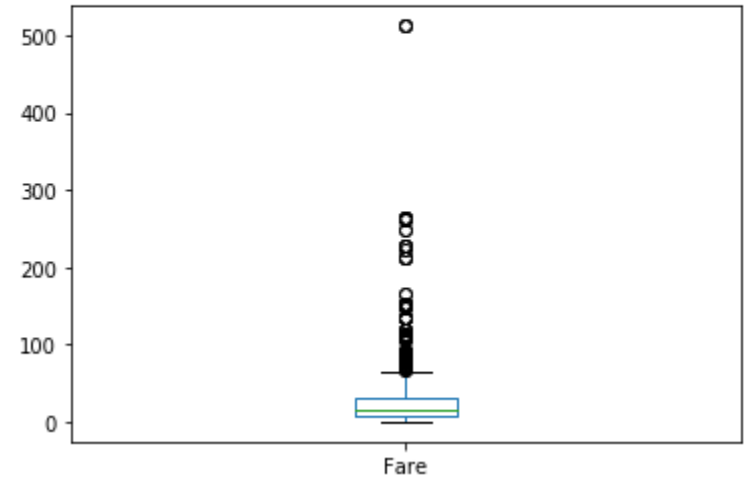
KIBERNUM

¿Qué es un valor atípico?

En el caso del valor del pasaje pagado del dataset Titanic, nótese que el rango mínimo máximo se encuentra entre 0 y 85 aproximadamente, y que cualquier valor por sobre el valor de 85 es considerado un punto atípico.

```
df['Fare'].plot(kind='box')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x211d4ee1f88>
```



¿Qué es un valor atípico?

Para calcular qué valores son atípicos en un conjunto de datos numéricos, existen varios criterios. A continuación se presenta el más frecuente de utilizar:

- Tomar el valor de la mediana, en este caso, es 14.4542
- Tomar valor del primer cuartil, en este caso, Q1 es 7.9104
- Tomar valor del tercer cuartil, este caso, Q3 es 31.0
- Calcular el rango intercuartil, es decir, tomamos IQR = Q3 – Q1
- Calcular el límite interno **superior** de la siguiente forma:
  LSUP = Q3 + 1.5 * IQR
- Calcular el límite interno **inferior** de la siguiente forma:
  LINF = Q1 – 1.5 * IQR
- Calcular los límites externos del conjunto de datos, de forma análoga pero considerando multiplicar *3 el rango intercuartil.

```
df['Fare'].describe()
```

```
count    891.000000
mean      32.204208
std       49.693429
min        0.000000
25%        7.910400
50%       14.454200
75%       31.000000
max      512.329200
Name: Fare, dtype: float64
```

¿Qué hacer con los valores atípicos?

Lo primero que debemos hacer es identificarlos, para después analizarlos de forma más detallada.

Recordemos que los valores atípicos podrían deberse a la data sucia, errores. Sin embargo, antes de comenzar a eliminarlos hay que considerar la visión de los analistas quienes deberán evaluar la conveniencia de su permanencia o bien eliminación.

Dudas y consultas

KIBERNUM

Creando Juntos Excelencia

Gracias

KIBERNUM