

▼ Data Analysis with Python

▶ Módulo 1

[] ↳ 21 celdas ocultas

▼ Módulo 2

▼ Missing values

Usar `dataframes.dropna()`:

`axis=0` --> retira toda la fila

`axis=1` --> retira toda la columna

```
df.dropna(subset=['X1'], axis=0, inplace=True) # Para modificar el dataframe utilizamos inpla
df                                              # Sin esta instrucción df no se modificará
```



	ID	X1	X2	X3	X4	X5	X6	X7	X8	X9	...	X15	X16	
0	1	20000	2.0	2.0	1.0	24.0	2.0	2.0	-1.0	-1.0	...	0.0	0.0	

Para remplazar datos faltantes

```

2      3      90000      2.0      2.0      2.0      34.0      0.0      0.0      0.0      0.0      ...      14331.0      14946.0      1554
mean = df['X3'].mean()
df['X3'].replace(np.nan, mean)

0      2.0
1      2.0
2      2.0
3      2.0
4      2.0
...
29995    3.0
29996    3.0
29997    2.0
29998    3.0
29999    2.0
Name: X3, Length: 30000, dtype: float64

```

▼ Data Formatting

Para hacer una conversión de formato de medida inglesa a medida internacional, podemos usar el siguiente código

`df["city_mpg"] = 235 / df[city_mpg]` --> En esta línea de código hacemos la conversión numérica
`df.rename(columns={"city_mpg": "city_L/100km"}, inplace=True)` --> En esta línea cambiamos el nombre de la columna y la reemplazamos en el data frame

Tipos de datos incorrectos

Para cambiar el tipo de datos que tenemos podemos utilizar las siguientes líneas de código

- Primero identificamos el tipo de dato


```
dataframe.dtypes()
```

- Posteriormente lo convertimos

```
dataframe.astype()
```

▼ Data Normalization


```
data = {'Income':[10000, 20000, 30000, 40000],
        'Age':[20, 21, 19, 18]}
df1 = pd.DataFrame(data)
df1
```

	Income	Age	
0	10000	20	
1	20000	21	
2	30000	19	
3	40000	18	

Simple feature scaling

$$X_{\text{new}} = X_{\text{old}} / X_{\text{max}}$$

```
df1["Income"] = df1["Income"]/df1["Income"].max()
df1["Age"] = df1["Age"]/df1["Age"].max()
df1
```

	Income	Age	
0	0.25	0.952381	
1	0.50	1.000000	
2	0.75	0.904762	
3	1.00	0.857143	

Min-Max

$$X_{\text{new}} = (X_{\text{old}} - X_{\text{min}}) / (X_{\text{max}} - X_{\text{min}})$$

```
df2 = df1.copy()
df2["Income"] = (df2["Income"] - df2['Income'].min()) / (df2['Income'].max() - df2['Income'].min())
df2["Age"] = (df2["Age"] - df2['Age'].min()) / (df2['Age'].max() - df2['Age'].min())
df2
```

Income**Age**

Z-score

$$X_{\text{new}} = (X_{\text{old}} - \mu) / \text{std dev}$$

```
df3 = df1.copy()
df3['Income'] = (df3['Income'] - df3['Income'].mean()) / (df3['Income'].std())
df3['Age'] = (df3['Age'] - df3['Age'].mean()) / (df3['Age'].std())
df3
```

Income**Age**

0	-1.161895	0.387298
1	-0.387298	1.161895
2	0.387298	-0.387298
3	1.161895	-1.161895

▼ Binning

```
data_bin = {'Price':[1000, 600, 100, 2500],
            'Category':['High', 'Medium', 'Low', 'High']}
df_ = pd.DataFrame(data_bin)
df_
```

Price**Category**

0	1000	High
1	600	Medium
2	100	Low
3	2500	High

```
bins = np.linspace(min(df_['Price']), max(df_['Price']), 4)
group_names = ['Low', 'Medium', 'High']
df_['Category'] = pd.cut(df_['Price'], bins, labels=group_names, include_lowest=True)
df_.hist(bins=3)
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f12ec8e2e50>]],
      dtype=object)
```



▼ Categorical variables into quantitive variables

0 500 1000 1500 2000 2500

```
data_cars = {'Car':['A', 'B', 'C', 'D'], 'Fuel':['gas', 'diesel', 'gas', 'gas']}
df_cars = pd.DataFrame(data_cars)
df_cars
```

	Car	Fuel	
0	A	gas	
1	B	diesel	
2	C	gas	
3	D	gas	

Con pandas podemos convertir variables categóricas en variables cuantitativas con la función `dummy`

```
Dummie = pd.get_dummies(df_cars['Fuel'])
Dummie
```

	diesel	gas	
0	0	1	
1	1	0	
2	0	1	
3	0	1	

Productos de pago de Colab - Cancelar contratos

✓ 0 s completado a las 19:50

