

UNIDAD

V1

**DIPLOMATURA EN PYTHON APLICADO A LA
CIENCIA DE DATOS**

Funciones útiles Pandas

Tips Pandas

Pandas es una biblioteca que nos permite el análisis y manipulación de datos. Proporciona numerosas funciones y métodos que agilizan el proceso de análisis y exploración de datos. Dentro de la amplia selección de funciones y métodos, algunos de ellos se utilizan con más frecuencia. Proporcionan una forma rápida de obtener una comprensión básica de los datos disponibles.

Para las siguientes funciones que se describen a continuación vamos a utilizar un dataset conocido que es la información de Diabetes.- El CSV está adjuntado en el apartado de varios.-

```
In [4]: import os
import pandas as pd
import numpy as npdiabetes
os.getcwd()
pd.read_csv("diabetes.csv")
```

Out[4]:

	6	148	72	35	0	33.6	0.627	50	1
0	1	85	66	29	0	26.6	0.351	31	0
1	8	183	64	0	0	23.3	0.672	32	1
2	1	89	66	23	94	28.1	0.167	21	0
3	0	137	40	35	168	43.1	2.288	33	1
4	5	116	74	0	0	25.6	0.201	30	0
...
762	10	101	76	48	180	32.9	0.171	63	0
763	2	122	70	27	0	36.8	0.340	27	0
764	5	121	72	23	112	26.2	0.245	30	0
765	1	126	60	0	0	30.1	0.349	47	1
766	1	93	70	31	0	30.4	0.315	23	0

767 rows × 9 columns

1. Función Head and Tail

Una vez que tenemos cargado los datos en nuestro dataframe si queremos ver de una manera rápida nuestros datos podemos utilizar la función Head y Tail, las cuales nos permiten mostrar la parte superior y la parte inferior.-

```
In [8]: diabetes.head()
```

```
Out[8]:
```

	6	148	72	35	0	33.6	0.627	50	1
0	1	85	66	29	0	26.6	0.351	31	0
1	8	183	64	0	0	23.3	0.672	32	1
2	1	89	66	23	94	28.1	0.167	21	0
3	0	137	40	35	168	43.1	2.288	33	1
4	5	116	74	0	0	25.6	0.201	30	0

```
In [ ]:
```

Se muestran 5 filas de forma predeterminada, pero podemos ajustarlo simplemente pasando el número de filas que nos gustaría mostrar. Por ejemplo, `diabetes.head(10)` mostrará las primeras 10 filas.

```
In [23]: diabetes.head()
#Por Defecto me muestra 5 filas - Puedo cambiarlo pasando como parámetro La cantidad
```

```
Out[23]:
```

	Embarazos	Glucosa	Presión Arterial	Grosor de pliegue de la piel del tríceps	Insulina sérica de 2 horas	Índice de masa Corporal	Función de pedigrí de diabetes	Edad	Clase - aparición de diabetes en un plazo de cinco años
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
In [15]: diabetes.head(10)
```

```
Out[15]:
```

	Cantidad de Embarazos	Glucosa	Presión Arterial	Grosor de pliegue de la piel del tríceps	Insulina sérica de 2 horas	Índice de masa Corporal	Función de pedigrí de diabetes	Edad	Clase - aparición de diabetes en un plazo de cinco años
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1

⇒ Ojo acá con los espacios en la cabecera, si queremos consultar por una columna en particular no dejar espacios en las descripciones

2. Función Nunique

Cuando se trabaja con datos categóricos o características que tienen valores discretos, es muy importante conocer el número de valores únicos. Es un paso esencial hacia la exploración de datos.

Una forma es usar la función `value_counts` que devuelve una serie de pandas con valores únicos en una columna y el número de apariciones de cada valor. La longitud de esta serie es el número de valores únicos.

```
In [24]: len(diabetes.Embarazos.value_counts())
```

```
Out[24]: 17
```

```
In [25]: diabetes.columns
```

```
In [26]: diabetes.Embarazos.nunique()
```

```
Out[26]: 17
```

Aplicando al todo el dataset

```
In [27]: diabetes.nunique()
```

```
Out[27]: Embarazos          17
         Glucosa           136
         Presión Arterial    47
         Grosor de pliegue de la piel del tríceps  51
         Insulina sérica de 2 horas  186
         Índice de masa Corporal  248
         Función de pedigrí de diabetes  517
         Edad              52
         Clase - aparición de diabetes en un plazo de cinco años  2
         dtype: int64
```

3. Funciones Dtypes y Astype

Necesitamos tener los valores almacenados en un tipo de datos apropiado. De lo contrario, podemos encontrar errores. Para conjuntos de datos grandes, el uso de la memoria se ve muy afectado por la selección correcta del tipo de datos. Por ejemplo, el tipo de datos "categórico" es más apropiado que el tipo de datos "objeto" para datos categóricos, especialmente cuando el número de categorías es mucho menor que el número de filas.

```
In [32]: diabetes.dtypes
```

```
Out[32]: Embarazos          int64
Glucosa          int64
Presión Arterial  int64
Grosor de pliegue de la piel del tríceps  int64
Insulina sérica de 2 horas  int64
Índice de masa Corporal    float64
Función de pedigrí de diabetes  float64
Edad              int64
Clase - aparición de diabetes en un plazo de cinco años  int64
dtype: object
```

En caso de querer cambiar un tipo de datos podemos usar la función `astype`.-

```
In [32]: diabetes.dtypes
```

```
Out[32]: Embarazos          int64
Glucosa          int64
Presión Arterial  int64
Grosor de pliegue de la piel del tríceps  int64
Insulina sérica de 2 horas  int64
Índice de masa Corporal    float64
Función de pedigrí de diabetes  float64
Edad              int64
Clase - aparición de diabetes en un plazo de cinco años  int64
dtype: object
```

```
In [45]: diabetes.Glucosa = diabetes.Glucosa.astype('float64')
```

```
In [46]: diabetes.dtypes
```

```
Out[46]: Embarazos          int64
Glucosa          float64
Presión Arterial  int64
GrosorDePliegueDeLaPiel  int64
InsulinaSéricaDe2HRS  int64
ÍndiceDeMasaCorporal    float64
FunciónDePedigríDeDiabetes  float64
Edad              int64
Clase              int64
dtype: object
```

4. Funciones Shape y Size

Shape se puede utilizar en matrices numpy, series de pandas y marcos de datos. Muestra el número de dimensiones así como el tamaño en cada dimensión.

Dado que los marcos de datos son bidimensionales, la forma que devuelve es el número de filas y columnas. Es una medida de la cantidad de datos que tenemos y una entrada clave para el proceso de análisis de datos.

Además, la proporción de filas y columnas es muy importante al diseñar e implementar un modelo de aprendizaje automático. Si no tenemos suficientes observaciones (filas) con respecto a las características (columnas), es posible que necesitemos aplicar algunas

técnicas de pre procesamiento, como la reducción de dimensionalidad o la extracción de características.

```
In [48]: diabetes.shape
```

```
Out[48]: (768, 9)
```

```
In [49]: diabetes.size
```

```
Out[49]: 6912
```

```
In [50]: diabetes.Embarazos
```

```
Out[50]: 0      6
         1      1
         2      8
         3      1
         4      0
         ..
        763    10
        764      2
        765      5
        766      1
        767      1
```

5. Función Describe

La función Describe brinda una descripción general rápida de las columnas numéricas al proporcionar estadísticas básicas como la media, la mediana y la desviación estándar.

```
In [28]: diabetes.describe()
```

```
Out[28]:
```

	Embarazos	Glucosa	Presión Arterial	Grosor de pliegue de la piel del tríceps	Insulina sérica de 2 horas	Índice de masa Corporal	Función de pedigrí de diabetes	Edad	Clase - aparición de diabetes en un plazo de cinco años
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

25%, 50% y 75% son los percentiles 50% también se conoce como la mediana, que es el valor en el medio cuando se ordenan todos los valores.

El 25% es el primer cuartil, por lo que el 25% de los valores están por debajo de este punto. Puede considerarlo como la mediana de la mitad inferior de una columna.

Del mismo modo, el 75% es el tercer cuartil.

Las estadísticas básicas nos brindan información valiosa. Por ejemplo, al comparar la media y la mediana (50%), comprendemos los valores atípicos. Si la media es mucho más alta que la mediana, hay valores atípicos en la parte superior.

Las estadísticas básicas también nos brindan una descripción general de la distribución de datos.

También puede seleccionar diferentes percentiles para mostrar mediante el parámetro de percentiles.

Los valores predeterminados son [.25, .50, .75]. Por ejemplo, percentiles = [1, .25, .5, .75, .9] mostrará percentiles de 10% y 90% además de los valores predeterminados.-

6. Función Isna

La function Isna es de gran utilidad porque nos permite analizar los valores faltantes. En nuestro ejemplo no tenemos porque es un set de datos limpio.-

Detectar los valores faltantes es clave ya que nos pueden ocasionar valores no deseables en los cálculos de precisión.-

La función Isna nos devuelve valores booleanos de True o False.-

diabetes.describe()

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False

Es un paso crítico para construir un proceso de análisis de datos sólido. Los valores faltantes deben ser de máxima prioridad, ya que tienen un efecto significativo en la precisión de cualquier análisis.

```
: diabetes.isna().any
```

```
: <bound method DataFrame.any of
0      False  False      False      False      False      False
1      False  False      False      False      False      False
2      False  False      False      False      False      False
3      False  False      False      False      False      False
4      False  False      False      False      False      False
..      ...    ...      ...      ...      ...      ...
763     False  False      False      False      False      False
764     False  False      False      False      False      False
765     False  False      False      False      False      False
766     False  False      False      False      False      False
767     False  False      False      False      False      False

      Embarazos  Glucosa  Presión Arterial  GrosorDePliegueDeLaPiel \
0      False      False      False      False
1      False      False      False      False
2      False      False      False      False
3      False      False      False      False
4      False      False      False      False
..      ...      ...      ...      ...
763     False      False      False      False
764     False      False      False      False
765     False      False      False      False
766     False      False      False      False
767     False      False      False      False

      InsulinaSéricaDe2HRS  ÍndiceDeMasaCorporal \
0      False      False
1      False      False
2      False      False
3      False      False
4      False      False
..      ...      ...
763     False      False
764     False      False
765     False      False
766     False      False
767     False      False
```

```
In [54]: diabetes.isna().sum()
```

```
Out[54]: Embarazos          0
          Glucosa            0
          Presión Arterial    0
          GrosorDePliegueDeLaPiel  0
          InsulinaSéricaDe2HRS  0
          ÍndiceDeMasaCorporal  0
          FunciónDePedigríDeDiabetes  0
          Edad              0
          Clase              0
          dtype: int64
```

```
In [ ]:
```

A menos que desee analizar el marco de datos celda por celda, isna debe combinarse con una agregación. isna (). any () indica si hay algún valor perdido en una columna, mientras que isna (). sum () devuelve el número de valores perdidos en las columnas.

El conjunto de datos que tenemos no tiene valores perdidos, pero este es un caso muy poco probable en la vida real.

IsNull(), isna(), notna() detectan valores faltantes.

Podemos querer reemplazar los datos nulos en este caso podemos utilizar la función replace()

Por ejemplo: diabetes.fillna(50).

7. Función Groupby

La función es una gran herramienta para explorar los datos. Facilita descubrir las relaciones subyacentes entre las variables.

¿Cómo funciona?

Agrupamos por una característica, luego agrega por una operación (Media, Suma, etc), luego podemos pedirle que lo ordene (Ascendente o Descendente).-

8. Función Sort

También podemos ordenar el resultado agregando la función sort_values:

Se clasifica en orden ascendente de forma predeterminada. Podemos cambiarlo estableciendo el parámetro ascendente como falso.


```
In [55]: diabetes[['Clase', 'Glucosa']].groupby('Clase').mean().sort_values('Clase')
```

```
Out[55]:
```

Glucosa	
Clase	
0	109.980000
1	141.257463

```
In [56]: diabetes[['Clase', 'Glucosa']].groupby('Clase').mean().sort_values('Clase', ascending=False)
```

```
Out[56]:
```

Glucosa	
Clase	
1	141.257463
0	109.980000

```
In [57]: diabetes[['Clase', 'Glucosa']].groupby('Clase').mean().sort_values('Clase', ascending=True)
```

```
Out[57]:
```

Glucosa	
Clase	
0	109.980000
1	141.257463