Python para Análisis de Datos

Módulo 04

Pandas

Estadísticas

Pandas provee numerosos métodos para realizar estadísticas sobre los datos tanto en series como en dataframes. Por defecto, estos métodos ignoran los missing values, no teniéndolos en cuenta a la hora de realizar los cálculos. Si se usa el parámetro `skipna = False` los valores `NaN` se propagan al resultado final.

Cuando se trabaja con un dataframe, estas funciones se calculan por columna y sólo las que tienen números y devuelven una serie.

Entre estos métodos podemos encontrar: `sum, mean, std, var, mode, median, count, min, max, prod`

Estadísticas

```
data.mean()
```

```
PassengerId    446.000000
Survived         0.383838
Pclass           2.308642
Age             29.699118
SibSp            0.523008
Parch            0.381594
Fare            32.204208
dtype: float64
```

```
# promedio y desviación standard de las edades
data["Age"].mean(), data["Age"].std()
```

```
(29.69911764705882, 14.526497332334044)
```

```
# promedio de edad de los pasajeros de primera clase
data.loc[data["Pclass"]==1, "Age"].mean()
```

```
38.233440860215055
```

```
# pomedio de edad de las mujeres de primera clase
data.loc[(data["Pclass"]==1) & (data["Sex"]=="female"), "Age"].mean()
```

```
34.61176470588235
```

Estadísticas

```
# Contar cantidad de varones
data.loc[data["Sex"]=="male", "Sex"].count()
```

```
577
```

```
# contar cantidad de varones
# usando que en la suma True vale 1
(data["Sex"]=="male").sum()
```

```
577
```

```
data["Age"].min(), data["Age"].max()
```

```
(0.42, 80.0)
```

```
# cantidad de elementos únicos
# en cada columna
data.nunique()
```

```
PassengerId    891
Survived         2
Pclass           3
Name           891
Sex              2
Age             88
SibSp            7
Parch            7
Ticket         681
Fare           248
Cabin          147
Embarked         3
dtype: int64
```

describe()

El método `describe` permite ver un resumen de las estadísticas principales de cada columna.

Cuando la columna es **numérica** devuelve: cantidad de elementos válidos, promedio, desviación standard, mínimo, máximo y cuartiles.

Cuando la columna **no es numérica** devuelve: cantidad de elementos válidos, cantidad de elementos únicos y el elemento más frecuente y su frecuencia.

Cuando el dataframe incluye **columnas numéricas y no numéricas,** el método sólo muestra el resumen de las columnas numéricas. Este comportamiento se puede controlar con los parámetros `include/exclude.`

describe()

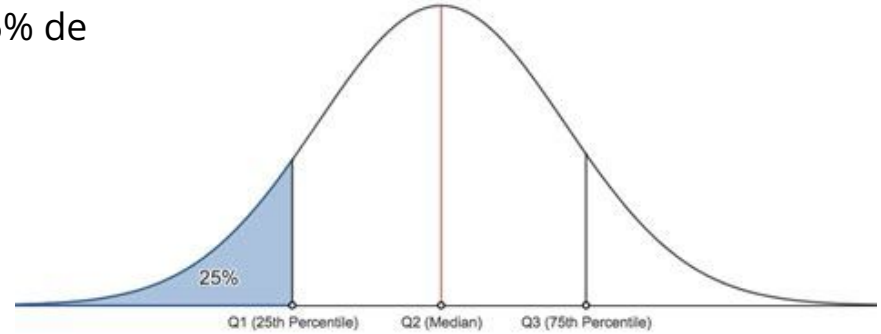
```
data.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Percentiles

Por defecto, se muestran los cuartiles, pero se puede controlar para que muestre cualquier percentil a través del parámetro `percentiles`.

Recordemos que el **percentil** es el valor para el cual un determinado porcentaje de los datos son menores que ese valor. Así, el percentil 25 es el valor por debajo del cual se encuentra el 25% de los datos.



aggregate()

El método `aggregate` (con el alias *agg*) permite hacer un resumen con las funciones de agregación que querramos. Se le puede pasar como argumentos un string con el nombre de la función o la función en sí (que puede incluir funciones de numpy).

En principio estas funciones se calculan sobre todas las columnas, lo que puede causar resultados inesperados. Por ejemplo, la función `sum` aplicada a strings, los concatena todos. Se pueden seleccionar las columnas numéricas con el método `select_dtypes`.

aggregate()

```
data.select_dtypes("number").aggregate(["sum", "min", np.max, "var"])
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
sum	397386.0	342.000000	2057.000000	21205.170000	466.000000	340.000000	28693.949300
min	1.0	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
amax	891.0	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200
var	66231.0	0.236772	0.699015	211.019125	1.216043	0.649728	2469.436846

value_counts()

El método `value_counts` para series permite encontrar una distribución de frecuencias de los valores únicos que contiene. Devuelve una serie cuyos índices son los valores únicos de la serie original y sus valores son las respectivas frecuencias.

El resultado está ordenado de mayor a menor, de modo que el primer elemento es el más frecuente.

Se puede hacer que se ordene en orden ascendente con el parámetro `ascending`.

El parámetro `normalize` permite obtener las proporciones de cada valor.

¡Muchas gracias!

¡Sigamos trabajando!