Introducción a Pandas

Rolando Salazar

[1] *Universidad de Sonora, Hermosillo, Sonora.*

February 28, 2019

En la actividad que corresponde a este reporte, se realiza una introducción a la biblioteca de python "pandas" la cual es de gran utilidad para el análisis de datos.

1 Introducción

En esta actividad (act. 3) se llevó a cabo la exploración de funciones de pandas para el análisis estadístico de un conjunto de datos. Se proporcionaron algunas funciones y algunas otras fueron buscadas personalmente. Esta vez se utilizaron datos metereológicos de la ciudad de Navojoa encontrados en el sitio web oficial del Servicio Metereológico Nacional.

2 Funciones de Pandas

Tras haber importado las librerias Pandas y Matplotlib, se ejecutaron los siguientes comandos en un archivo de Jupyter Notebooks y se obtuvieron los distintos resultados mostrados a continuación.

2.1 Función read_csv

```
df.read_csv('datos.txt', skiprows=0, sep='\s')
```

Esta función sirve para que nuestro conjunto de datos quede en una sola variable que almacena a todos en un marco.

2.2 head

```
df.head()
```

Muestra las primeras 5 filas del entorno de datos.

	FECHA	PRECIP	EVAP	TMAX	TMIN
0	17/12/1967	0.0	Nulo	19.0	Nulo
1	18/12/1967	0.0	2	21.0	6
2	19/12/1967	0.0	2.7	20.0	9
3	20/12/1967	0.0	7.7	23.0	12
4	21/12/1967	0.0	3.4	21.0	11

Figure 1: *Resultado de la función head.*

	FECHA	PRECIP	EVAP	TMAX	TMIN
369	23/05/1969	0.0	10.2	38.0	17
370	24/05/1969	0.0	10.4	36.0	16
371	25/05/1969	0.0	4.4	36.0	16.5
372	26/05/1969	0.0	11.8	37.0	15
373	27/05/1969	0.0	9.8	37.5	15

Figure 2: *Resultado de la función tail.*

2.3 tail

```
df.tail()
```

Muestra las últimas cinco filas de los datos.

2.4 dtypes

```
df.dtypes
```

Simplemente nos dice que tipo de variable es cada una de las columnas del marco de datos.

2.5 mean

```
df.mean()
```

Arroja una lista con las medias de las columnos de datos.

2.6 std

```
df.std()
```

Cálcula y muestra la variación estándar de cada columna.

2.7 median

```
df.median()
```

Obtiene las medianas de cada columna.

2.8 max

```
df.max()
```

Cálcula el valor máximo de cada columna.

2.9 min

```
df.min()
```

Lo contrario al anterior.

2.10 describe

```
df.describe()
```

Esta función realiza un análisis exploratorio de datos más detallado, el cual incluye muchas de las funciones anteriores.

2.11 dropna

```
df.dropna()
```

Esta función elimina todas las filas que contengan datos nulos o NA.

2.12 plot.hist

```
df.plot.hist()
```

Con este comando creamos un histograma por cada columna.

2.13 value_counts

```
df.["TMAX"].value_counts()
```

Cuenta el número de filas con cada valor único de la variable TMAX.

	PRECIP	TMAX
count	374.000000	374.000000
mean	0.310963	29.438503
std	2.239066	6.348373
min	0.000000	15.000000
25%	0.000000	24.000000
50%	0.000000	29.000000
75%	0.000000	34.375000
max	27.000000	41.500000

Figure 3: *Resultado de la función describe.*

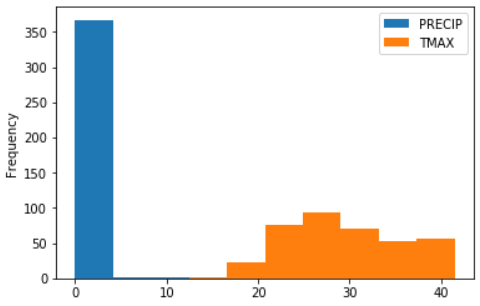

Figure 4: *Resultado de la función plot.hist.*

	FECHA	PRECIP	EVAP	TMAX	TMIN
364	18/05/1969	0.0	9.6	36.0	14
361	15/05/1969	0.0	6.3	31.0	12.5
261	04/02/1969	0.0	1	25.0	4
287	02/03/1969	0.0	6.9	25.0	6
69	24/02/1968	0.0	2.8	29.0	11
31	17/01/1968	0.0	2.8	25.0	8
304	19/03/1969	0.0	5.7	30.0	5.5
163	28/09/1968	0.0	7.7	40.0	22
111	07/05/1968	0.0	9.3	34.5	11
197	02/12/1968	0.0	2.6	26.0	Nulo
127	23/05/1968	0.0	11.4	34.0	16
295	10/03/1969	0.0	2.7	22.5	8.5
172	07/10/1968	0.0	7.7	39.0	22
64	19/02/1968	0.0	4.4	29.0	10
348	02/05/1969	0.0	9.1	29.5	15
23	09/01/1968	0.0	1.5	23.0	5.5

Figure 5: *Resultado de la función sample(frac=0.5).*

	FECHA	PRECIP	EVAP	TMAX	TMIN
230	04/01/1969	0.0	1.8	24.5	7
28	14/01/1968	0.0	3.5	22.0	8
340	24/04/1969	0.0	9.7	35.5	8.5
351	05/05/1969	0.0	6.2	25.0	13
84	10/04/1968	0.0	6.8	30.0	11
337	21/04/1969	0.0	9.6	35.0	9
209	14/12/1968	0.0	4.5	18.0	7
27	13/01/1968	0.0	3.3	23.0	10
322	06/04/1969	0.0	8.1	33.5	8
63	18/02/1968	0.0	4.2	27.0	9

Figure 6: *Resultado de la función sample(n=10).*

2.14 len

```
len(df)
```

Muestra la cantidad de filas.

2.15 nunique

```
df["TMAX"].nunique()
```

Numero de valores distintos en la columna.

2.16 sample con el atributo frac

```
df.sample(frac=0.5)
```

Fracción de filas seleccionadas al azar.

2.17 sample con atributo n

```
df.sample(n=10)
```

Selecciona n filas al azar.

2.18 iloc

```
df.iloc(40:45)
```

Selecciona las filas indicadas.

	FECHA	PRECIP	EVAP	TMAX	TMIN
40	26/01/1968	8.0	2.1	24.0	13
41	27/01/1968	0.0	1.1	26.0	15
42	28/01/1968	0.0	1.9	25.0	14
43	29/01/1968	0.0	0.9	23.0	13
44	30/01/1968	0.0	0.9	26.0	14

Figure 7: *Resultado de la función iloc.*

	FECHA	PRECIP	EVAP	TMAX	TMIN
198	03/12/1968	0.0	6.1	15.0	5
21	07/01/1968	0.0	1	17.0	11
270	13/02/1969	27.0	2	17.5	7
209	14/12/1968	0.0	4.5	18.0	7
255	29/01/1969	0.0	2.1	18.0	13
265	08/02/1969	0.0	1.1	18.0	6
215	20/12/1968	0.0	4	18.5	2.5
256	30/01/1969	0.0	1.8	18.5	8
0	17/12/1967	0.0	Nulo	19.0	Nulo
299	14/03/1969	0.0	4.8	19.5	5.5
262	05/02/1969	0.4	1.9	19.5	10
276	19/02/1969	0.0	3.5	20.0	8

Figure 8: *Resultado de la función sort_values.*

2.19 sort_values

```
df.sort_values('TMAX')
```

Ordena los valores de las filas de menor a mayor de la columna seleccionada.

2.20 rename

```
df.rename(columns={"PRECIP":"PREC"})
```

Reescribe el nombre de una columna.

3 Conclusiones

Se pueden combinar estos comandos para manipular los datos como