01.Lectura_datos_clima

October 29, 2022

1 Pandas para el manejo de datos tabulares

En esta sección usaremos **Pandas**: documentación, repositorio del código para leer datos tabulares.

Una de las mejores opciones para trabajar con datos tabulares en Python es usar Python Data Analysis Library (alias `Pandas`). La biblioteca `Pandas` provee estructuras de datos, genera gráficos de alta calidad con matplotlib y se integra de buena forma con otras bibliotecas que usan arrays de NumPy.

1.1 Datos

Los datos que vamos a trabajar fueron descargados desde el SENAMHI para la estación del Ñaña cerca de Lima.

1.2 Importar librerias

[ ]:
```
import pandas as pd
```

1.3 Leer archivo

[ ]:
```
data = pd.read_csv('../../data/qc00000543.txt')
data
```

[ ]:
```
           1964 2 1 0 25.6 17
0       1964 2 2 0 25.2 16.4
1          1964 2 3 0 26 18
2        1964 2 4 0 -99.9 18
3        1964 2 5 0 -99.9 17
4      1964 2 6 0 -99.9 18.2
…                      …
18257    2014 1 27 0 27.8 17
18258  2014 1 28 0 27.2 17.8
18259  2014 1 29 0 27.6 17.8
18260  2014 1 30 0 27.8 17.4
18261    2014 1 31 0 26.6 17

[18262 rows x 1 columns]
```

Podemo usar **`delimiter`** para separar las columnas.

[ ]:
```
data = pd.read_csv('../../data/qc00000543.txt', delimiter=' ')
data
```

[ ]:
```
        1964  2   1    0  25.6    17
0       1964  2   2  0.0  25.2  16.4
1       1964  2   3  0.0  26.0  18.0
2       1964  2   4  0.0 -99.9  18.0
3       1964  2   5  0.0 -99.9  17.0
4       1964  2   6  0.0 -99.9  18.2
…       …  ..  ..  …   …   …
18257  2014  1  27  0.0  27.8  17.0
18258  2014  1  28  0.0  27.2  17.8
18259  2014  1  29  0.0  27.6  17.8
18260  2014  1  30  0.0  27.8  17.4
18261  2014  1  31  0.0  26.6  17.0

[18262 rows x 6 columns]
```

El archivo no tiene encabezado. Entonces podemos usar **delimiter** para indicar que el archivo no tiene encabezado.

[ ]:
```
data = pd.read_csv('../../data/qc00000543.txt', delimiter=' ', header=None)
data
```

[ ]:
```
           0  1   2    3     4     5
0       1964  2   1  0.0  25.6  17.0
1       1964  2   2  0.0  25.2  16.4
2       1964  2   3  0.0  26.0  18.0
3       1964  2   4  0.0 -99.9  18.0
4       1964  2   5  0.0 -99.9  17.0
…       …  ..  ..  …   …   …
18258  2014  1  27  0.0  27.8  17.0
18259  2014  1  28  0.0  27.2  17.8
18260  2014  1  29  0.0  27.6  17.8
18261  2014  1  30  0.0  27.8  17.4
18262  2014  1  31  0.0  26.6  17.0

[18263 rows x 6 columns]
```

Los datos tienen valores perdidos y estos estan declarados como -99.9. Podemos usar

[ ]:
```
data = pd.read_csv('../../data/qc00000543.txt', delimiter=' ', header=None,␣
↪na_values=-99.9)
data
```

[ ]:
```
           0  1   2    3     4     5
0       1964  2   1  0.0  25.6  17.0
1       1964  2   2  0.0  25.2  16.4
```

```
2       1964  2   3  0.0  26.0  18.0
3       1964  2   4  0.0   NaN  18.0
4       1964  2   5  0.0   NaN  17.0
…       …  ..  ..  …    …    …
18258   2014  1  27  0.0  27.8  17.0
18259   2014  1  28  0.0  27.2  17.8
18260   2014  1  29  0.0  27.6  17.8
18261   2014  1  30  0.0  27.8  17.4
18262   2014  1  31  0.0  26.6  17.0

[18263 rows x 6 columns]
```

Ahora vamos a agragar en nombre de las variables para cada columna. Para eso es importante saber que representa cada columna.

[ ]:
```
data.columns = ['Year', 'Month', 'Day', 'Prec', 'Tmax', 'Tmin']
data
```

[ ]:
```
        Year  Month  Day  Prec  Tmax  Tmin
0       1964      2    1   0.0  25.6  17.0
1       1964      2    2   0.0  25.2  16.4
2       1964      2    3   0.0  26.0  18.0
3       1964      2    4   0.0   NaN  18.0
4       1964      2    5   0.0   NaN  17.0
…       …     …  …   …   …    …    …
18258   2014      1   27   0.0  27.8  17.0
18259   2014      1   28   0.0  27.2  17.8
18260   2014      1   29   0.0  27.6  17.8
18261   2014      1   30   0.0  27.8  17.4
18262   2014      1   31   0.0  26.6  17.0

[18263 rows x 6 columns]
```

Ahora vamos a gener una variable con formato datetime. Para eso usaremos **to_dateime**.

[ ]:
```
# Esto es un ejemplo de como generar una variable con formato tiempo.
pd.to_datetime(data[['Year', 'Month', 'Day']])
```

[ ]:
```
0       1964-02-01
1       1964-02-02
2       1964-02-03
3       1964-02-04
4       1964-02-05
           …
18258   2014-01-27
18259   2014-01-28
18260   2014-01-29
18261   2014-01-30
```

```
18262   2014-01-31
Length: 18263, dtype: datetime64[ns]
```

Vamos a ingresar esta variable en nuestro dataframe como el index. Esto hara que python pueda trabajar como una serie temporal.

[ ]:
```
data.index = pd.to_datetime(data[['Year', 'Month', 'Day']])
data
```

[ ]:

	Year	Month	Day	Prec	Tmax	Tmin
1964-02-01	1964	2	1	0.0	25.6	17.0
1964-02-02	1964	2	2	0.0	25.2	16.4
1964-02-03	1964	2	3	0.0	26.0	18.0
1964-02-04	1964	2	4	0.0	NaN	18.0
1964-02-05	1964	2	5	0.0	NaN	17.0
…	…	…	…	…	…	…
2014-01-27	2014	1	27	0.0	27.8	17.0
2014-01-28	2014	1	28	0.0	27.2	17.8
2014-01-29	2014	1	29	0.0	27.6	17.8
2014-01-30	2014	1	30	0.0	27.8	17.4
2014-01-31	2014	1	31	0.0	26.6	17.0

[18263 rows x 6 columns]

Podemos colocar un nombre a nuestro Index.

[ ]:
```
data.index.name = 'Datetime'
data
```

[ ]:

Datetime	Year	Month	Day	Prec	Tmax	Tmin
1964-02-01	1964	2	1	0.0	25.6	17.0
1964-02-02	1964	2	2	0.0	25.2	16.4
1964-02-03	1964	2	3	0.0	26.0	18.0
1964-02-04	1964	2	4	0.0	NaN	18.0
1964-02-05	1964	2	5	0.0	NaN	17.0
…	…	…	…	…	…	…
2014-01-27	2014	1	27	0.0	27.8	17.0
2014-01-28	2014	1	28	0.0	27.2	17.8
2014-01-29	2014	1	29	0.0	27.6	17.8
2014-01-30	2014	1	30	0.0	27.8	17.4
2014-01-31	2014	1	31	0.0	26.6	17.0

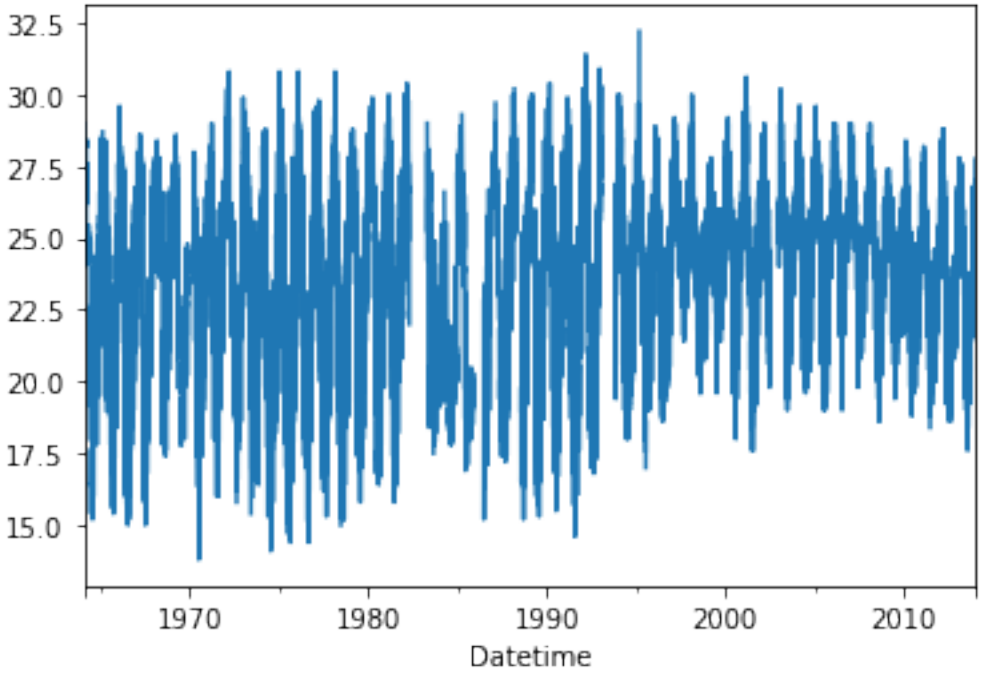
[18263 rows x 6 columns]

De esta forma ya estamos lito para usar nuestro dataframe como una serie de tiempo. Ahora podemos aprovechar una gran variedad de funcione disponibles en **pandas** para usar con nuetro dataframe. Por exmplo, podemos usar la función **plot** para hacer un grafico rapido de nuestros

datos.

[ ]:
```
data['Tmax'].plot()
```

[ ]:
```
<AxesSubplot:xlabel='Datetime'>
```

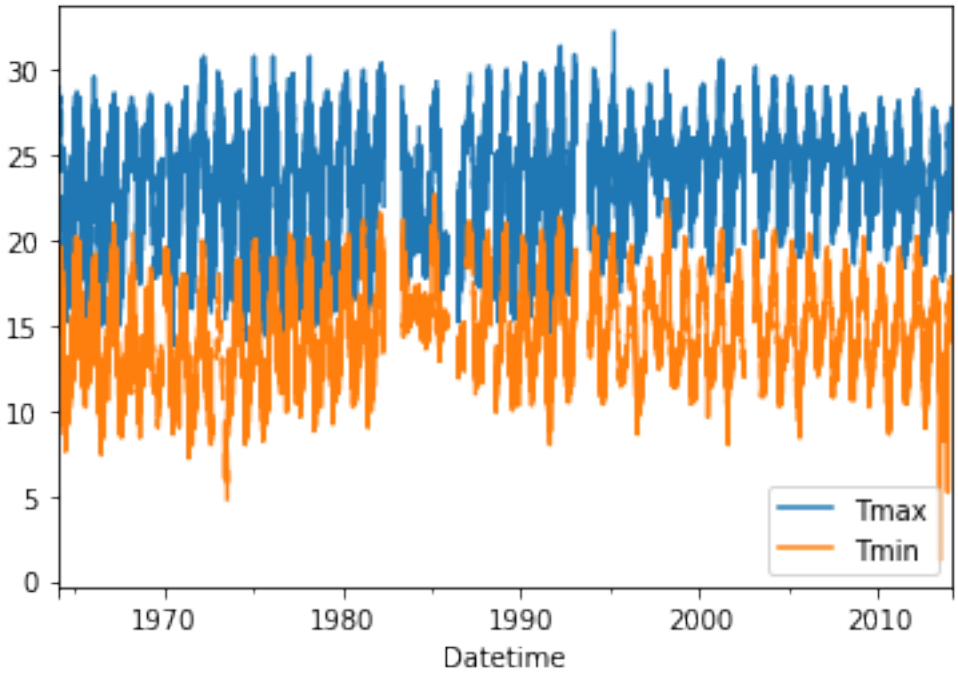


Podemo explorar nuestros datos en la figura que generamos.

Podemos seleccionar más de dos variables para hacer una figura.

[ ]:
```
data[['Tmax', 'Tmin']].plot()
```

[ ]:
```
<AxesSubplot:xlabel='Datetime'>
```


30
25
20
15
10
5
0
1970
1980
1990
2000
2010
Datetime
Tmax
Tmin