Actividad 3

Introducción a la biblioteca Pandas de Python

Jorge Benz Olguín Aguilar

División de Ciencias Exactas, Departamento de Física

Universidad de Sonora

23 de febrero de 2019

1. Introducción

En Computación y Ciencia de datos, pandas es una biblioteca de software escrita como extensión de NumPy para manipulación y análisis de datos para el lenguaje de programación Python. En particular, ofrece estructuras de datos y operaciones para manipular tablas numéricas y series temporales. Es un software libre distribuido bajo la licencia BSD versión tres cláusulas. El de la biblioteca deriva del término "datos de panel", término de econometría que designa datos que combinan una dimensión temporal con otra dimensión transversal.[1]

2. Estación Hermosillo

Para la práctica número tres utilizamos los datos de la estación de Hermosillo del servicio meteorológico nacional. Descargamos los datos en archivo tipo txt para su posterior manipulación con el entorno de programación Jupyter. Desde la estación de Jupyter llamamos al archivo utilizando la función pd.read_csv(), como lo podemos ver a continuación

```
data = pd.read_csv("Datos_Hermosillo_editado.txt"
,sep="\s+",skipfooter=1,skiprows=1,engine='python'
,names=['Fecha', 'Precip', 'Evap', 'Tmax', 'Tmin'])
```

En la tabla 1 podemos ver un resumen de las funciones y métodos utilizados para realizar la actividad 3, algunos son propios de python y otros de las bibliotecas de pandas y numpy. No lo

mencionamos antes, pero, estas bibliotecas tienen que ser llamadas antes de trabajar con ellas. Lo haremos de la siguiente manera

```
import pandas as pd
import numpy as np
```

Vamos a utilizar las funciones y métodos mencionados en la la tabla 1 para analizar los datos, podremos obtener información como: cuales son los meses más lluviosos, cuales son los más fríos o los mas cálidos, años mas secos y años más húmedos, etc.

¿Cómo le podrás determinar cuáles son los meses más lluviosos?

Cuando se bajaron los datos estos estaban organizados en cinco columnas y una de ellas nos proporcionaba la fecha de cada medición; día, mes y año. Esto nos llevo a tener que crear dos columnas nuevas, una de ellas sería para los meses y la otra para los años. Además los datos no tenían asignado su correspondiente tipo: enteros, flotante, etc. por lo que tuvimos que asignarle un valor numérico a las columnas correspondientes.

```
# Crear columnas con Año y Mes extraídos de la fecha
data['Año'] = data['NFecha'].dt.year
data['Mes'] = data['NFecha'].dt.month
data.head()

#Cambiamos el tipo de variable de las columnas a flotante
data[['Precip','Evap','Tmax','Tmin']] = data[['Precip','Evap','Tmax','Tmin']] \
.apply(pd.to_numeric, errors='coerce')
```

Determinar que meses son los mas lluviosos de una colección de mediciones lo podríamos determinar de dos formas: la primera seria hacerlo mes por mes, lo que significaría mucho mas trabajo, esto es mas código; la segunda, utilizando un loop que recorra todos los meses, cuatro o cinco lineas de código serian suficiente.

```
#utilizando un loop
total=0.0
for i in range(12):
PrecipMensual = data['Precip'][data['Mes']==[i+1]].sum()/NumA
total=total+PrecipMensual
print("Mes", i+1,":", np.round(PrecipMensual,decimals=2), "mm", ", Acumulada:",
np.round(total, decimals=2), "mm")
```

Función	Acción
data.head()	Nos permite ver los primeros cinco renglones de nuestro archivo de datos
data.tail()	Muestra los últimos cinco renglones del archivo de datos
data.dtypes	Nos permite conocer con que tipo de variables estamos trabajando: enteros, flotante, etc
data.mean()	Saca el promedio de las columnas del archivo
data.std()	Saca la desviación estándar de cada una de las columnas del archivo
data.median()	Nos permite obtener la mediana de las columnas del archivo
data.max()	Nos regresa el valor mas grande de cada columna
data.min()	Nos retorna el valor mas pequeño de cada una de las columnas del archivo
data.describe()	Obtenemos un resumen estadístico del archivo por columna: máximo, mínimo, desviación estándar, percentil(25,50 y 75), promedio y una cuanta de los elementos de la columna
len()	Regresa el número de elementos en una lista
unique()	Solo toma en cuenta los elementos no repetidos de una lista
sum()	suma todos los elementos de una lista; en nuestro caso, una columna
np.round()	Nos sirve para redondear los decimales de los elementos de una matriz
range()	Utilizando los parámetros del método establecemos un rango de actuación
count()	Método que regresa el número de veces que aparece una subcadena en una cadena dada
drop()	remueve filas y columnas
apply()	Aplica una funcion a cada columna y fila

Tabla 1: Funciones y métodos básicos

¿Cuáles son los meses más fríos y cuáles son los más cálidos?
De la misma forma que lo hicimos para los meses más lluviosos lo hacemos para los meses más fríos y más cálidos, esto es, utilizando un loop. El loop recorre los meses y todos los datos de la columna temperatura mínima o máxima, los datos son extraídos para el correspondiente mes, sumados y divididos entre el numero de datos encontrados.

```
# Para calcular los meses más frios y mas calidos utilizamos dos loops


for i in range(12):
TminPromMensual = data[data['Mes']==i+1]['Tmin'].sum()
/data[data['Mes'] ==i+1]['Tmin'].count()
print("Tmin Mes", i+1,":",
np.round(TminPromMensual, decimals=2), "°C")


print(" ")


for i in range(12):
TmaxPromMensual = data[data['Mes']==i+1]['Tmax'].sum()
/data[data['Mes'] ==i+1]['Tmax'].count()
print("Tmax Mes", i+1,":",
np.round(TmaxPromMensual, decimals=2), "°C")
```

¿Cómo ha venido siendo la temperatura mensual promedio en los últimos 20 años?
Se saco un promedio entre las temperaturas máximas y mínimas de los últimos 20 años, por mes. Todo dentro de un loop, lo cual como ya lo mencionamos nos evita un poco mas de trabajo.

```
# la temperatura mensual promedio en los últimos 20 años
for i in range(12):
TminPromMensual = data[data['Mes']==i+1]['Tmin'].sum()/data[data['Mes'] ==i+1]['Tmin'].c
TmaxPromMensual = data[data['Mes']==i+1]['Tmax'].sum()/data[data['Mes'] ==i+1]['Tmax'].c
TpromMensual = (TminPromMensual+TmaxPromMensual)*(1/2)
```

Estas son solo algunos ejemplos de lo que podemos hacer con un archivo de datos. Las herramientas de análisis que maneja la biblioteca pandas así como numpy son de las mejores que se pueden encontrar en forma gratuita e incluso dentro de las de paga.

Referencias

[1] Pandas. (2018, 27 de noviembre). Wikipedia, La enciclopedia libre. Fecha de consulta: 04:32, febrero 19, 2019