Universidad de Sonora

División de Ciencias Exactas y Naturales

Física Computacional

Visualizando datos con Pandas y Matplotlib

Autor:
Carlos Alí Medina Leal
Profesor:
Carlos Lizárraga Celaya

1. Un breve resumen...

En esta actividad se aprende a graficar usando Python/Pandas con las columnas de datos de la preferencia del usuario. También se aprende a juntar gráficas en una sola y acotar cada una.

2. Introducción

En la siguiente actividad entraremos más de lleno al proceso de representar visualmente los datos para obtener información relevante .ºculta.enstre éstos, así como obtener datos descriptivos sobre el comportamiento de los datos.

Originalmente, para visualización en Python se utilizaba la biblioteca Matplotlib. Hoy existen otras herramientas alternativas como lo son Bokeh, ggplot (adaptación del paquete ggplot2 de R) y otros. En este caso, usaremos el Matplotlib y el Pyplot.

3. Desarrollo

Primero, le damos valores a las variables e importamos las librerías o paquetes que se utilizarán, con el siguiente código ya antes visto:

```
import pandas as pd
import matplotlib as plt
import numpy as np
import matplotlib.pyplot as mplt
df = pd.read_csv("/home/camedina/Computacional1/Computacional1/Actividad4/
FiltroDatos23ASD.csv")
```

Los datos que se usaron en esta actividad son de un solo día, específicamente el 23 de agosto y la estación que se utilizó fue San Diego (72293), como la pasada actividad.

En esta ocasión generaremos tres gráficas:

1. Presión (hPa) vs. Altura (m)
2. Temperatura (ºC) vs. Altura (m)
3. Temperatura de Rocío (DWPT ºC) vs. Altura (m)
4. Temperatura y Temperatura de Rocío en una sola gráfica.

Empecemos por la gráfica de la presión, que nos ayudará a generar las demás siguiendo el mismo proceso, pero antes de graficar, se necesita saber el nombre exacto de cada columna, con el comando "*df.columns*".

Ahora, se usará el siguiente código:

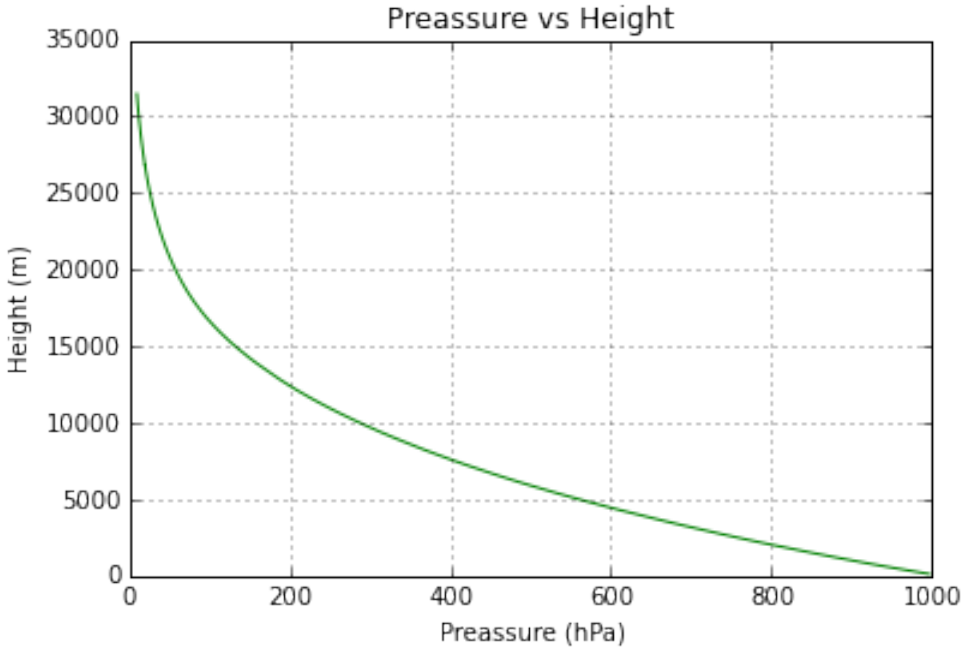
```
mplt.title('Preassure vs Height')
mplt.xlabel('Preassure (hPa)')
mplt.ylabel('Height (m)')
mplt.grid(True)

mplt.plot(df_complete[u'PRES'],df_complete[u'   HGHT   '], c='g')
```

Lo que hace el primer renglón del código es ponerle nombre a la gráfica, el segundo y el tercero les pone nombres a los ejes x y y respectivamente. El cuarto renglón traza una cuadrícula en el plano y el último es la gráfica. Para la gráfica se escribe el comando "*plot* " y entre paréntesis se escribe lo que utilizas para el eje x, después para el eje y, y finalmente con el comando " $c =' g'$ "señalé que quiero la línea color verde.

(NOTA: Recordar ejecutar el comando "%matplotlib inline" para que las gráficas no salgan en ventanas extras)

El resultado es el siguiente:



Después, con casi el mismo proceso, se generan las otras gráficas, usando:

```
mplt.title('Temperature vs Height')
mplt.xlabel('Temperature (C)')
mplt.ylabel('Height (m)')
mplt.grid(True)

mplt.plot(df_complete[u' TEMP  '],df_complete[u'  HGHT  '],c='aqua')
```

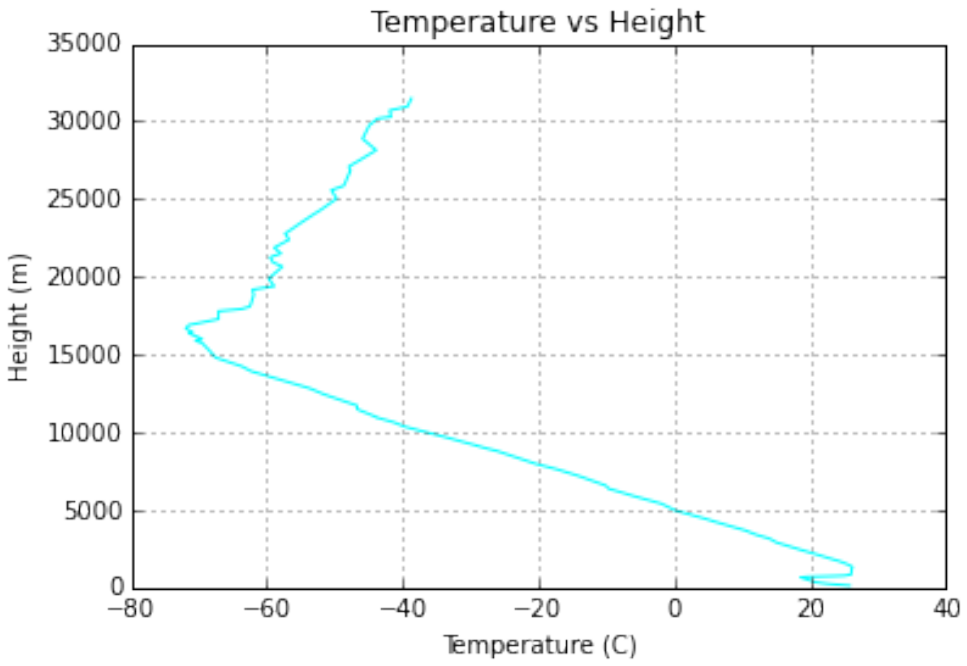
Para la gráfica de la temperatura contra la altura, y:

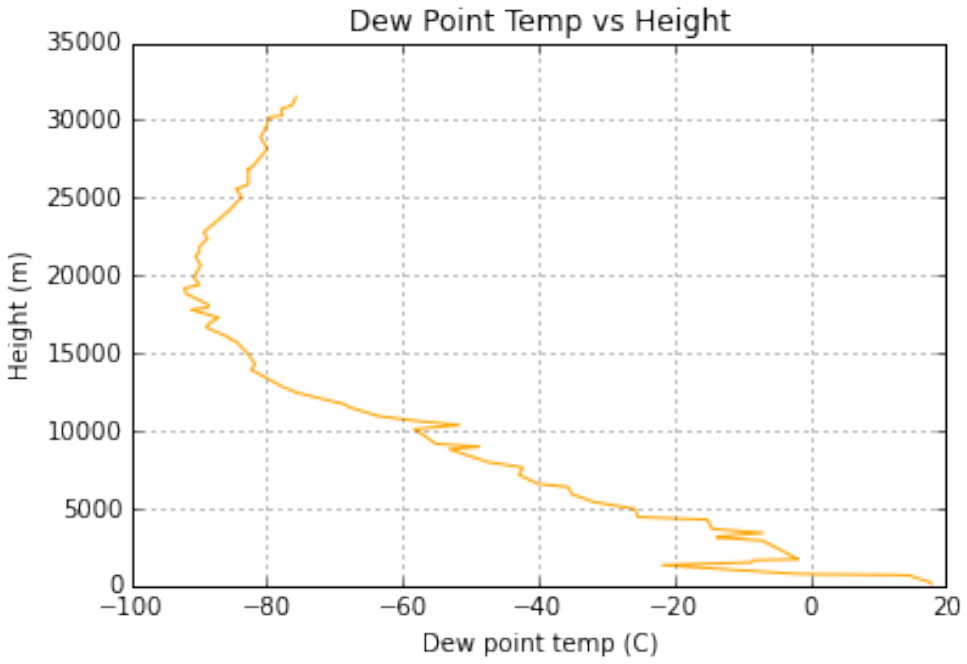
```
mplt.title('Dew Point Temp vs Height')
mplt.xlabel('Dew point temp (C)')
mplt.ylabel('Height (m)')
mplt.grid(True)

mplt.plot(df_complete[u' DWPT '],df_complete[u'  HGHT  '],c='orange')
```

Para la gráfica de la temperatura de rocío contra la altura.

Ambas gráficas resultan de la siguiente manera:



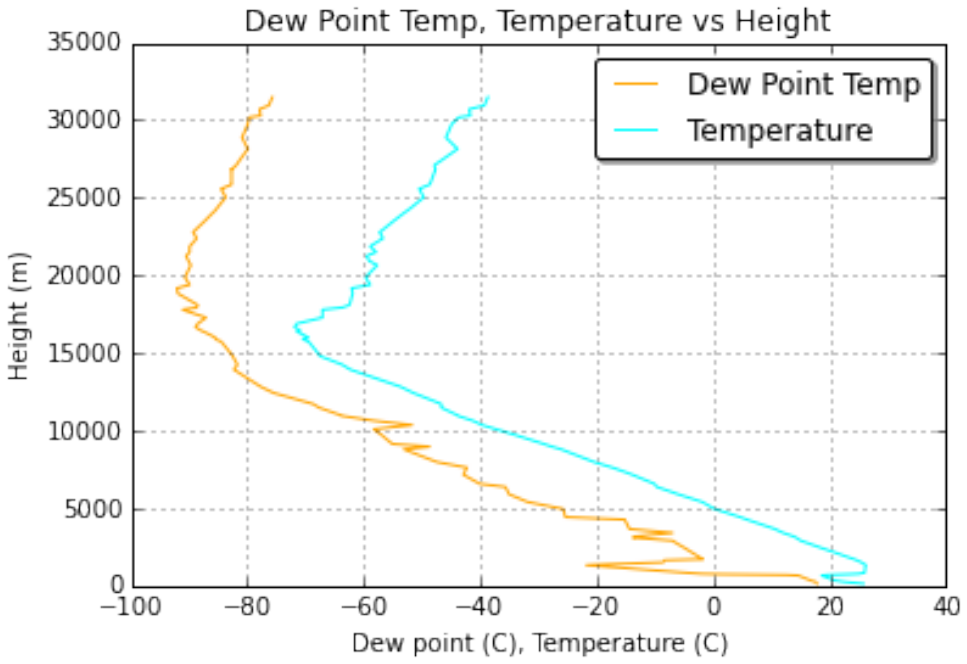


Ahora, para el último punto de la actividad, juntamos las dos gráficas, y no es algo tan complicado de hacer, simplemente se escribe otra gráfica en el mismo código y por el mismo comando "%matplotlib inline", se grafican juntas, resultando el siguiente código:

```
mplt.title('Dew Point Temp, Temperature vs Height')
mplt.xlabel('Dew point (C), Temperature (C)')
mplt.ylabel('Height (m)')
mplt.grid(True)

mplt.plot(df_complete[u' DWPT '],df_complete[u'   HGHT  '], 'orange',label="Dew Poi
mplt.plot(df_complete[u' TEMP  '],df_complete[u'   HGHT  '], 'aqua',label="Temperat
mplt.legend(fancybox=True,shadow=True)
```

Esto genera la siguiente gráfica:



Como se puede ver, en esta gráfica hay un cuadro de simbología para cada gráfica, esto se logra con el comando $label = "nombredelinea"$ en el paréntesis del plot, y el comando $mplt.legend(fancybox = True, shadow = True)$, el cual crea una caja y le añade sombra.

En esta actividad observamos que en Python/Pandas se tienen herramientas bastante útiles con las cuales se pueden analizar datos de manera accesible y relativamente fácil, así como generar gráficas que nos ayudan a visualizar mejor el comportamiento de datos, lo cual nos será muy útil en nuestra vida como estudiantes y profesionales en el ámbito científico.

4. Bibliografía

*U*niversity of Wyoming/ College of Engineering/ Department of Atmospheric Science : $http://weather.uwyo.edu/upperair/sounding.html$, Recuperado el 26 de Septiembre del 2016.

*P*ydata/ Python Data Analysis Library : $http://pandas.pydata.org/pandas-docs/stable/tutorials.html$, Recuperado el 27 de Septiembre del 2016.