Reporte sobre la Actividad 7

García Parra Pedro

Abril 2019

El objetivo principal de esta actividad fue comparar dos bibliotecas similares de python: **matplotlib** y **seaborn**. Estas bibliotecas en general sirven para realizar gráficas y analizar visualmente datos. La actividad nos pide realizar dos gráficas de heatmap con datos de correlación utilizando ambas bibliotecas. Comenzamos leyendo el archivo de datos, como éste contenía caracteres especiales se requirió el uso de el engine de python.

```
df = pd.read_csv("meteo-nogal-09.csv",engine = "python")
```

El documento contedia varias columnas que no nos aportaban ninguna imformación llamadas *unnamed* seguido de un número; para deshacerme de ellas utilicé otra libreria llamada **re** que me permitía usar expreciones regulares, así obtuve el nombre exacto de cada columna para despues eliminarlas con la función `dataframe.drop()` de **pandas**.
Una vez tenía las columnas exactas que ocupaba prosedí a obtener la correlación de cada columna con cada otra columna. Para ello utilicé la función de pandas `dataframe.corr()` que lo hace automaticamente y regresa un dataframe con esos valores.
Las *figuras* 1 y 2 muestran los mismos mapas de calor, pero la manera de realizarlos es muy diferente; con seaborn se requirió una sola línea de código para crear la gráfica e incluso se pudieron añadir líneas que separan los cuadros muy fácilmente:

```
sb.heatmap(corr,cmap="viridis",robust=True,square=True,
           linewidths=.01)
```

Pero con matplotlib fue más complicado porque muchas cosas se tuvieron que hacer *a mano*, el código fue el siguiente:

```
fig, ax = plt.subplots()
ax.set_xticks(np.arange(len(corr)))
ax.set_yticks(np.arange(len(corr)))

ax.set_xticklabels(corr.columns)
ax.set_yticklabels(corr.columns)

plt.setp(ax.get_xticklabels(), rotation=90,ha="right",
         rotation_mode="anchor")


plt.imshow(corr, cmap='viridis',interpolation='nearest')
plt.colorbar()
```

La actividad también pedía que se graficaran las columnas donde su correlación fuese mayor al 60 %; al pensar en como hacer ésto encontré dos problemas:

- La diagonal cumple con la condición de ser mayor al 60 % ya que ésta es siempre del 100 %
- Los datos de correlación estan duplicados usando la diagonal como espejo

Si graficaba sin pensar en estos problemas iba a crear gráficas que no tenian significancia y gráficas duplicadas. Para solucionar ésto primero creé una función que me corta un array cualquiera de un punto inicial a uno final:

```
def array(arr,st,fin):
    arra = []
    for i in range(st,fin):
        arra.append(arr[i])
    return arra
```

Creé esta función con la idea de utilizarla para cortar el array de las correlaciones utilizando un solo lado de la diagonal.
Una vez creada es una simple cuestión de utlizar un doble loop utilizando una variable i que recorra todas las columnas y otra variable j que recorra las columnas *recortadas* por la función y evitar cuando la i y la j sean iguales; el código es el siguiente:

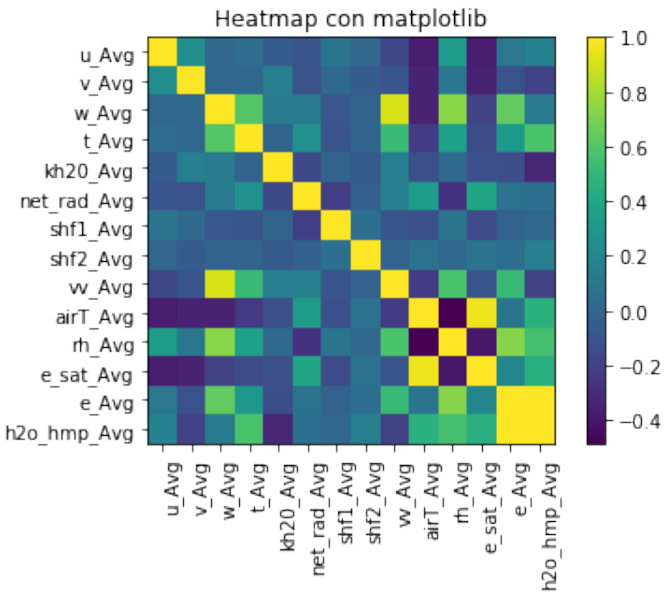

Figura 1: Heatmap realizado con la biblioteca de matplotlib

```
Cols = []
n = 0
for i in corr.index:
    for j in array(corr.index,n,len(corr)):
        if(abs(corr[i][j]) > 0.6 and i != j):
            Cols.append([i,j])
    n=n+1
```

Tras realizar ésto, tendremos un array de dos dimensiones llamado Cols en el que tendremos cada una de los pares de columnas que hay que graficar. Las *figuras* 3 y 4 muestran las gráficas con la correlación mayor y la menor.

Como conclusión puedo decir que utilizar seaborn fue mucho más fácil que matplotlib ya que seaborn tiene más funciones que realizan las tareas por tí, dicho ésto tambien es bueno saber que es lo que estas haciendo y no solamente escribir funciones que realizan las acciones por *magia* pero para un primer acercamiento y para principiantes o genete que solamente quiere los resultados es mucho mejor seaborn.

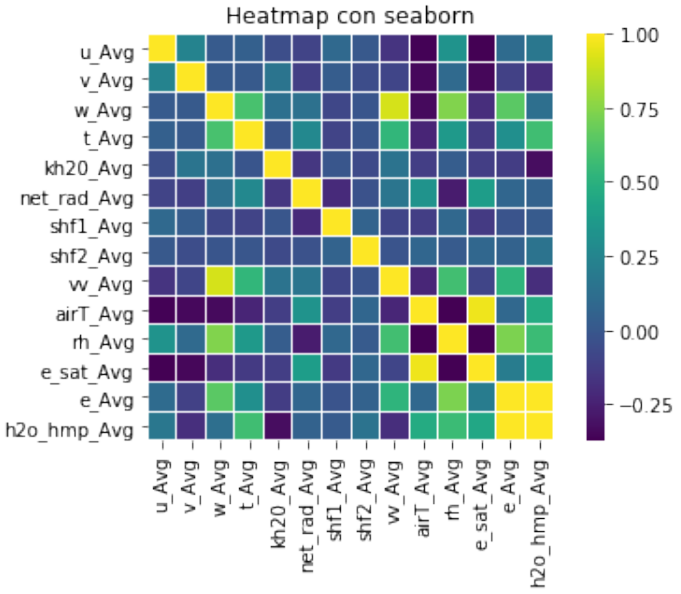

Figura 2: Heatmap realizado con la biblioteca de seaborn

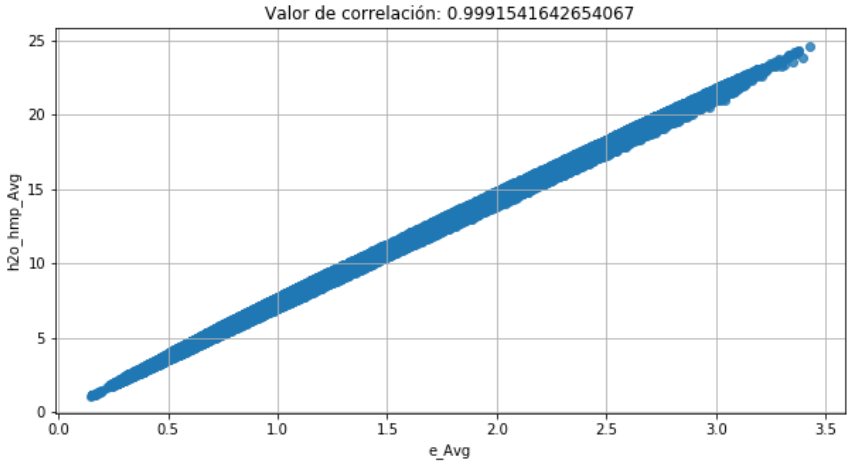

Figura 3: Gráfica de e Avg contra h2o hmp Avg con correlación lineal de 60.13 %

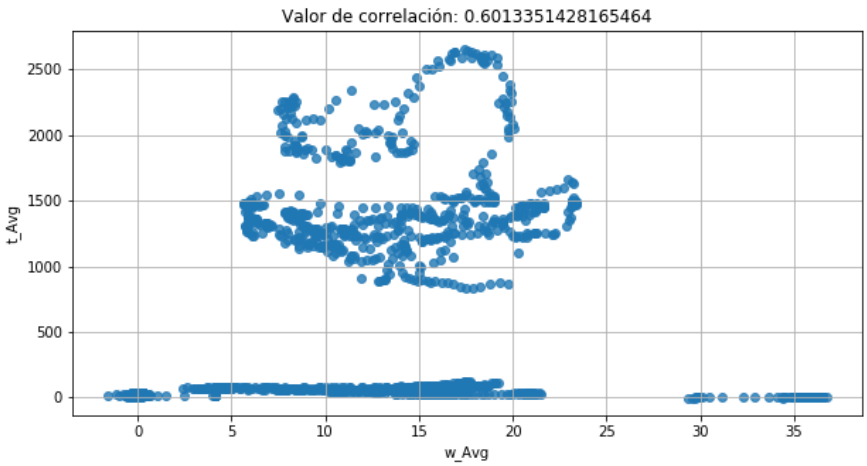

Figura 4: Gráfica de w Avg contra t Avg con correlación lineal de 99.91 %