Universidad de Sonora

Licenciatura en Física

Física Computacional I

Evaluación I

Alumno:
José Gabriel Navarro I.

Profesor:
Carlos Lizarraga Celaya

08 de Marzo de 2018

1 Análisis de datos

Primeramente se descargaron los datos correspondientes a la práctica en la página del curso. Una vez realizado esto, se observaron los contenidos del archivo para saber que datos eliminar. En este caso, se elimino un desfase que había con los tiempos, uno siendo a las 12:45 y otro a las 11:15. Esto no se realizo con emacs. Se realizo utilizando los comandos drop y skiprow al leer el archivo en Jupyter Notebook. También, como se pasaron los primeros renglones, se le dieron un nuevo nombre a las columnas.

```
#Leemos el primer archivo y quitamos el renglon sobrante.
df = pd.read_csv("sargento201117.csv", skiprows=2, header=None, names=['Num','Date','AbPr','Temp','WL']
)
df=df[:-1]
df.head()
```

	Num	Date	AbPr	Temp	WL
0	1	10/26/2017 13:00:00	105.612	24.448	-0.150
1	2	10/26/2017 13:15:00	105.513	24.351	-0.160
2	3	10/26/2017 13:30:00	105.433	24.351	-0.168
3	4	10/26/2017 13:45:00	105.385	24.351	-0.173
4	5	10/26/2017 14:00:00	105.321	24.351	-0.179

```
#Leemos el segundo archivo y quitamos el renglon sobrante.
dfS = pd.read_csv("sargentosalinidad201117.csv", skiprows=3, header=None, names=['Num','Date','CHR','Te
mp','SC','Sal'])
dfS.head()
```

	Num	Date	CHR	Temp	SC	Sal
0	2	10/26/2017 13:00:00	54525.5	24.91	54622.1	36.1588
1	3	10/26/2017 13:15:00	54525.5	24.82	54719.0	36.2311
2	4	10/26/2017 13:30:00	54525.5	24.76	54783.8	36.2794
3	5	10/26/2017 13:45:00	54525.5	24.75	54794.6	36.2875
4	6	10/26/2017 14:00:00	54525.5	24.73	54816.2	36.3036

Una vez echo esto, cambiamos el tipo de dato de la columna de fecha para ambos data frames, ya que estas fueron leídas como objetos. Esto se hizo mediante el uso de la librería datetime:

```
#Cambiamos el tipo de dato de la Fecha y creamos una nueva columna
df['NDate'] = pd.to_datetime(df['Date'], format='%m/%d/%Y %H:%M:%S')
dfS['NDate'] = pd.to_datetime(dfS['Date'], format='%m/%d/%Y %H:%M:%S')
```

```
df.head()
```

	Num	Date	AbPr	Temp	WL	NDate
0	1	10/26/2017 13:00:00	105.612	24.448	-0.150	2017-10-26 13:00:00
1	2	10/26/2017 13:15:00	105.513	24.351	-0.160	2017-10-26 13:15:00
2	3	10/26/2017 13:30:00	105.433	24.351	-0.168	2017-10-26 13:30:00
3	4	10/26/2017 13:45:00	105.385	24.351	-0.173	2017-10-26 13:45:00
4	5	10/26/2017 14:00:00	105.321	24.351	-0.179	2017-10-26 14:00:00

Con esto, ya es posible realizar las graficas que se requieren para la evaluación. Sin embargo, para mayor facilidad al graficar algunas de ellas, se crearon nuevos dataframes,

en donde solo se contienen las columnas necesarias para su graficación, esto se hizo mediante la creación de un nuevo dataframe seleccionando las columnas requeridas de los archivos:

```
#Creamos archivos para las graficas de Pearson
df_SalT = dfS[dfS.columns[5:6]]
df_SalT['Temp'] = dfS['Temp']
df_SalT.head()
```

	Sal	Temp
0	36.1588	24.91
1	36.2311	24.82
2	36.2794	24.76
3	36.2875	24.75
4	36.3036	24.73

De esta manera, se procedió a realizar las graficas correspondientes a la evaluación.

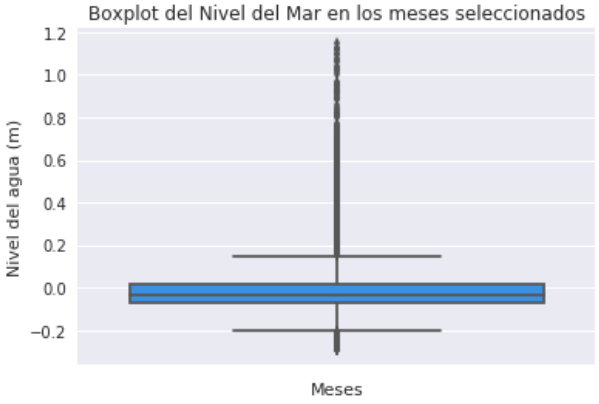
2 Análisis de resultados

En esta sección, se presentan las distintas graficas con su interpretación respectiva que se pedía en la evaluación.

Con la ayuda de la biblioteca Seaborn, por favor crea un gráfica de caja (boxplot) para visualizar la variabilidad de los datos de Febrero:

a) Nivel de mar (metros)

```
#Grafica Nivel del Mar contra el Tiempo (De caja)
ax = sns.boxplot(y="WL", data=df, color="dodgerblue")
ax.set_xlabel('Meses')
ax.set_ylabel('Nivel del agua (m)')
ax.set_title('Boxplot del Nivel del Mar en los meses seleccionados')
plt.show()
```



Para realizar esta gráfica, solamente se coloco en el eje de las 'y' el dato a analizar, sin separarlo en los distintos meses para estudiarlo de una mejor manera. En esta gráfica podemos observar como existe muchos puntos sesgados, es decir, fuera de la caja, la media y cuartiles están muy cerca de 0, causando que los datos mayores a estos sean sesgados (que son la mayoria). Esto puede observarse también en la descripción de los datos:

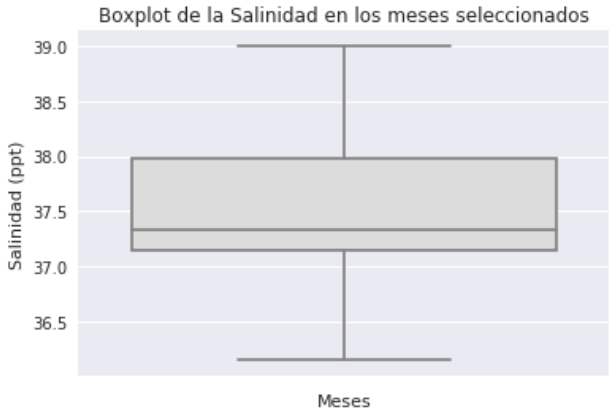
```
df.describe()
```

	Num	AbPr	Temp	WL
count	2394.000000	2394.000000	2394.000000	2394.000000
mean	1197.500000	107.430007	23.120883	0.030863
std	691.232595	2.371844	0.563555	0.235974
min	1.000000	104.229000	21.760000	-0.288000
25%	599.250000	106.407000	22.525000	-0.071000
50%	1197.500000	106.764000	23.388000	-0.035000
75%	1795.750000	107.305000	23.484000	0.018750
max	2394.000000	118.641000	24.448000	1.146000

En donde la mediana esta en 0.030 para WL (Water Level), y lo cuartiles es -0.07 y 0.018. Por lo cual muchos de los datos estan sesgados.

b) Salinidad (Partes por mil - ppt)

```
#Grafica Salinidad contra el Tiempo (De caja)
ax = sns.boxplot(y="Sal", data=dfS, color="gainsboro")
ax.set_xlabel('Meses')
ax.set_ylabel('Salinidad (ppt)')
ax.set_title('Boxplot de la Salinidad en los meses seleccionados')
plt.show()
```



Para graficar esta y el resto de las graficas de caja se realizo el mismo proceso que en la gráfica anterior, colocando el dato a estudiar en el eje y. En el caso de la salinidad, podemos observar que ninguno de los datos están sesgados:

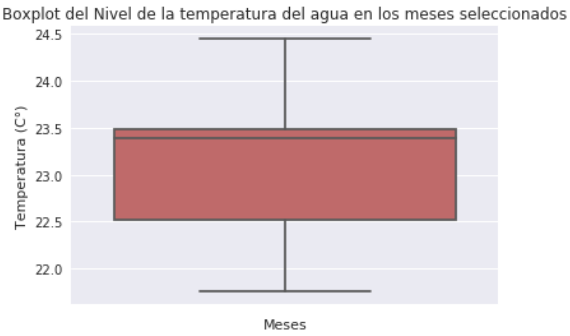
```
dfS.describe()
```

	Num	CHR	Temp	SC	Sal
count	2394.000000	2394.000000	2394.000000	2394.000000	2394.000000
mean	1198.500000	54524.972807	23.316646	56386.831662	37.479737
std	691.232595	11.876669	0.547033	619.501987	0.464974
min	2.000000	54105.700000	21.490000	54622.100000	36.158800
25%	600.250000	54525.500000	22.730000	55949.700000	37.151400
50%	1198.500000	54525.500000	23.490000	56185.600000	37.328300
75%	1796.750000	54525.500000	23.700000	57053.700000	37.980300
max	2395.000000	54525.500000	24.910000	58398.700000	38.994200

Podemos observar como el valor máximo y mínimo, (36.15 y 38.99 respectivamente), están incluidos dentro de la línea de la caja.

c) Temperatura de Agua (°C)

```
#Grafica Temperatura contra el Tiempo (De caja)
ax = sns.boxplot(y="Temp", data=df, color="indianred")
ax.set_xlabel('Meses')
ax.set_ylabel('Temperatura (C°)')
ax.set_title('Boxplot del Nivel de la temperatura del agua en los meses seleccionados')
plt.show()
```



De igual manera que la gráfica anterior, podemos observar como ninguno de sus datos están sesgados, pero su media esta mas cargada al tercer cuartil. Esto se puede observar con la tabla de la función describe:

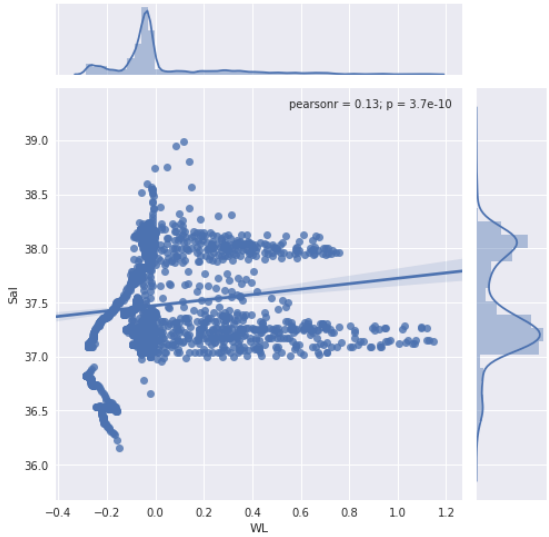
```
df.describe()
```

	Num	AbPr	Temp	WL
count	2394.000000	2394.000000	2394.000000	2394.000000
mean	1197.500000	107.430007	23.120883	0.030863
std	691.232595	2.371844	0.563555	0.235974
min	1.000000	104.229000	21.760000	-0.288000
25%	599.250000	106.407000	22.525000	-0.071000
50%	1197.500000	106.764000	23.388000	-0.035000
75%	1795.750000	107.305000	23.484000	0.018750
max	2394.000000	118.641000	24.448000	1.146000

De nuevo con la ayuda de Seaborn, también explora si hay una correlación de Pearson entre cada pareja de variables:

a) Nivel de mar-Salinidad

```
#Graficamos la primera grafica de Pearson, de Nivel de Mar con Salinidad
sns.set(style="darkgrid", color_codes=True)

g = sns.jointplot("WL", "Sal", data=df_SalNM , kind="reg",
                  color="b", size=7)
plt.show(g)
```

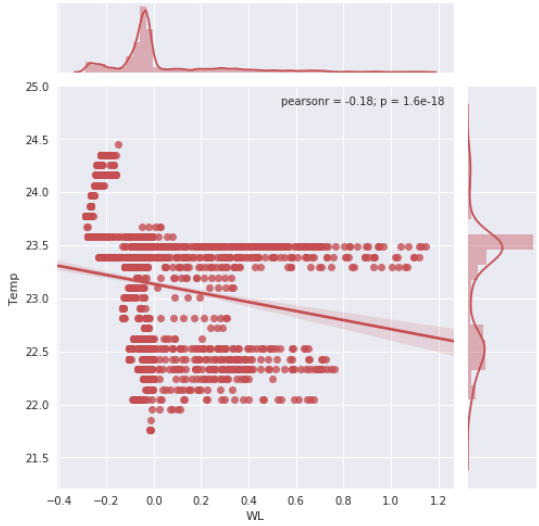


Esta gráfica se realizó como se hizo anteriormente en clases, utilizando la libreria seaborn, indicando las dos variables a comparar, en este caso Nivel de mar y Salinidad. Es en estas graficas que se utiliza los nuevos dataframes creados anteriormente.

En ella podemos observar un coeficiente de relación lineal de 0.13, con un factor de "no-relación" muy pequeño, lo cual nos dice que si hay poco de relación lineal entre estas dos variables.

b) Nivel de mar-Temperatura del agua

```
#Graficamos la segunda grafica de Pearson, de Nivel de Mar con Temperatura
sns.set(style="darkgrid", color_codes=True)

g = sns.jointplot("WL", "Temp", data=df_NMT, kind="reg",
                  color="r", size=7)
plt.show(g)
```



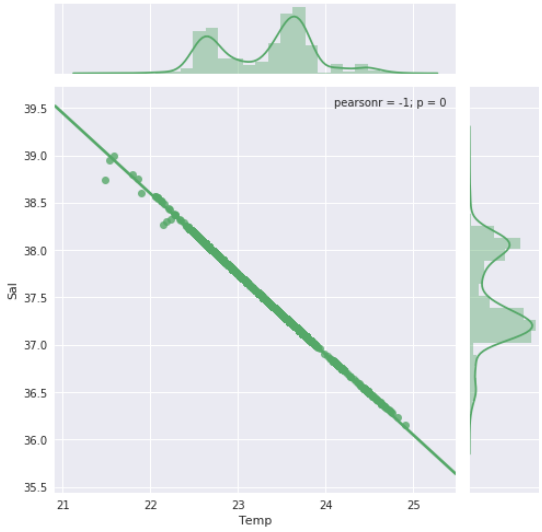
De igual manera, esta gráfica se realizo utilizando la librería seaborn, indicando los datos a usar, y de donde extraemos esta información. Esta vez se relaciona el nivel de mar con la temperatura del agua, que, como podemos observar, esta gráfica tiene una relación parecida a la anterior, con un valor de 0.13, pero esta vez negativo, indicando una recta hacia abajo.

c) Salinidad-Temperatura del agua

Por último, se muestra la gráfica que relaciona a la salinidad y la temperatura del agua, donde como se puede observar tienen un coeficiente de relación de -1. Eso indica que la relación entre temperatura y salinidad es casi seguro.

Al revisar estos datos y compararlos con los otros datos obtenidos de temperatura, se llego auna grafica muy parecida, con unos puntos mas dispersos pero con un coeficiente igual, de -1.

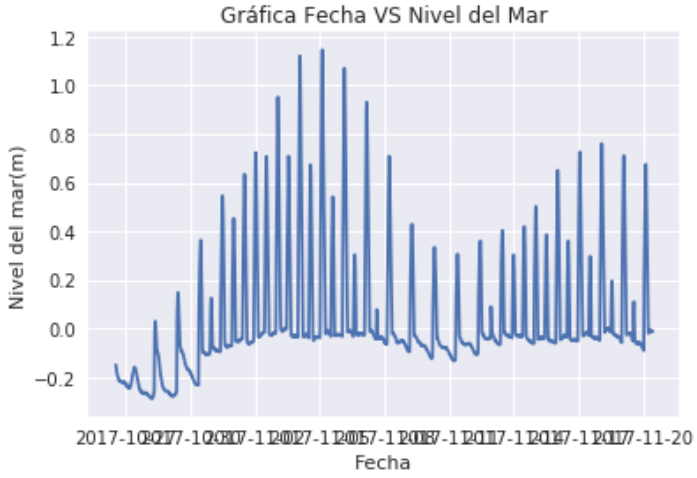
```
#Graficamos la segunda grafica de Pearson, de Salinindad con Temperatura
sns.set(style="darkgrid", color_codes=True)

g = sns.jointplot("Temp", "Sal", data=df_SalT, kind="reg",
                  color="g", size=7)
plt.show(g)
```



Con la ayuda de Matplotlib, realize ahora 3 gráficas independientes de las variables:

a) Nivel del mar

```
plt.plot_date(x=df.NDate, y=df.WL, fmt="b-")
plt.title("Gráfica Fecha VS Nivel del Mar")
plt.xlabel("Fecha")
plt.ylabel("Nivel del mar(m)")
plt.grid(True)
plt.show()
```



Esta gráfica fueron de las primeras que se realizaron al inicio del semestre, en donde se indica el tipo de gráfica que es (date), seguido por los datos a graficar (en este caso el nivel del mar), en donde fmt permite cambiar el color de la gráfica. Como se puede observar, los picos mas altos están entre el día 5 de noviembre.

b) Salinidad

En esta gráfica y la siguiente se realizo el mismo proceso que la anterior, seleccionando los datos y poniéndolos en contra de las fechas. Esta es la salinidad contra el tiempo:

```
#Graficamos Tiempo VS Salinidad
plt.plot_date(x=dfS.NDate, y=dfS.Sal, fmt="r-")
plt.title("Gráfica Fecha VS Salinidad")
plt.xlabel("Fecha")
plt.ylabel("Salininidad (ppt)")
plt.grid(True)
plt.show()
```

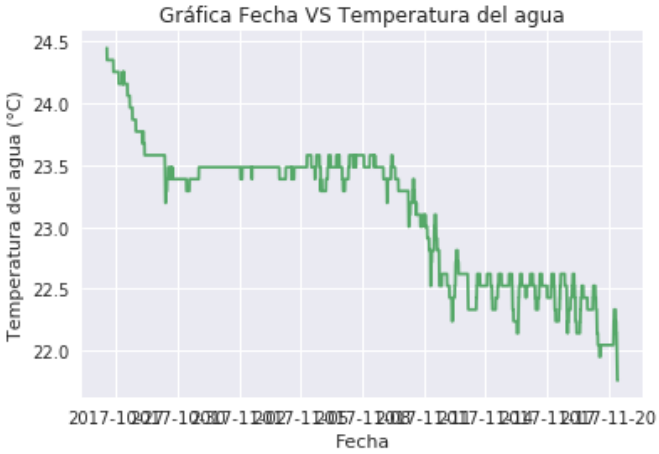
Como podemos notar, en los primeros datos la cantidad era la menor de todos los registros, pero es en estos mismos días que aumenta y toma su valor máximo.

c) Temperatura del agua

Por último se presenta la gráfica de la Temperatura con respecto el tiempo, en donde como podemos observar esta baja conforme el tiempo avanza.

Los datos están tomados en Octubre y en Noviembre, y como en Noviembre hace mas frío que Octubre, esto puede explicar el hecho de porque al avanzar el tiempo la temperatura del agua es cada vez menor. Sin embargo, esta temperatura no es exactamente muy grande, si observamos las tablas del comando describe, podemos ver como la mínima temperatura registrada es de 21.76°C y la máxima es de 24.44°C.

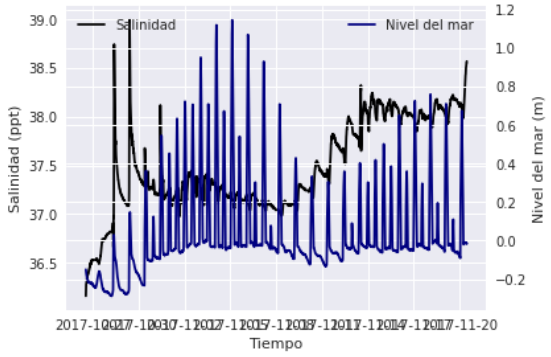
```
#Graficamos Tiempo VS Temperatura del Agua
plt.plot_date(x=df.NDate, y=df.Temp, fmt="g-")
plt.title("Gráfica Fecha VS Temperatura del agua")
plt.xlabel("Fecha")
plt.ylabel("Temperatura del agua (°C)")
plt.grid(True)
plt.show()
```



Enseguida, de igual forma, produce gráficas superpuestas con doble eje vertical (izquierda, derecha):

a) Nivel de mar y Salinidad

```
#Graficamos Salinidad y Nivel del mar vs Tiempo
fig, ax1 = plt.subplots()
tiem=df['NDate']
sal=dfS.Sal
NM=df.WL
ax1.plot(tiem,sal,"black", label='Salinidad'); plt.legend(loc='upper left')
ax1.set_xlabel('Tiempo')
ax1.set_ylabel('Salinidad (ppt)')
ax2 = ax1.twinx()
ax2.plot(tmp, NM , "navy", label='Nivel del mar'); plt.legend(loc='best')
ax2.set_ylabel('Nivel del mar (m)')
fig.tight_layout()
plt.show()
```

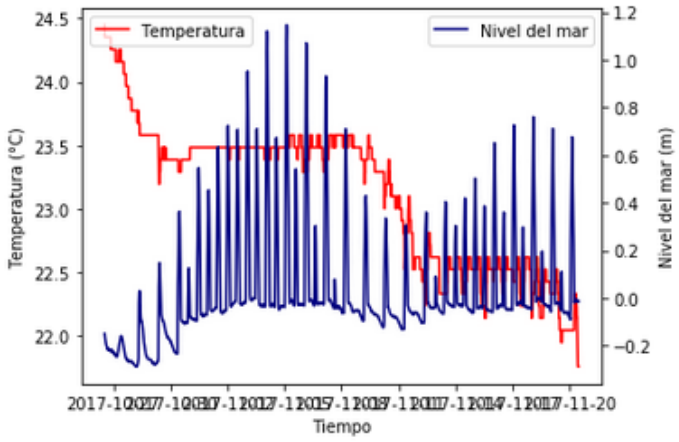


En esta gráfica se utilizo un nuevo concepto de graficas que no se había visto en clases. Se utilizan los "subplots", que estos pueden ser colocados posteriormente en una figura, o "fig". De esta manera es posible graficar dos lineas con datos distintos en y, pero el mismo eje x. Solo es cuestión de definir de donde se tomaran los datos y definir ambas lineas que se van a graficar.

Parece que al analizar la gráfica, podemos observar que al bajar mas el nivel del mar la salinidad aumenta, esto se puede ver mas claro en los primeras fechas. Esto se puede observar mejor en el siguiente paso de la evaluación.

b) Nivel de mar y Temperatura.

```
#Grafica Temperatura y Nivel del Mar vs Tiempo
fig, ax1 = plt.subplots()
tiem=df['NDate']
Temp=df.Temp
NM=df.WL
ax1.plot(tiem,Temp,"red", label='Temperatura'); plt.legend(loc='upper left')
ax1.set_xlabel('Tiempo')
ax1.set_ylabel('Temperatura (°C)')
ax2 = ax1.twinx()
ax2.plot(tiem, NM , "navy", label='Nivel del mar'); plt.legend(loc='best')
ax2.set_ylabel('Nivel del mar (m)')
fig.tight_layout()
plt.show()
```

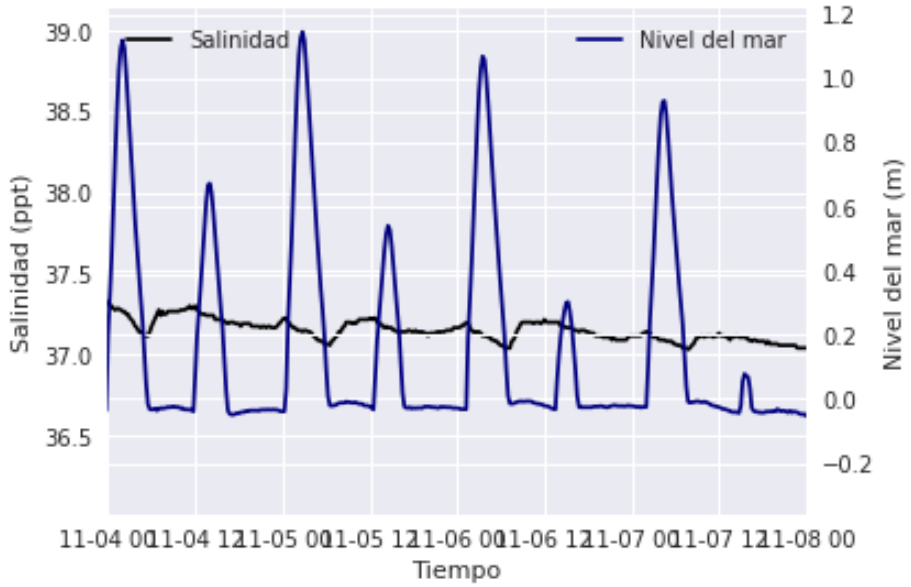


Esta gráfica se realizo de la misma manera que la anterior, tomando esta vez los datos de Fecha, Temperatura del agua y el nivel del mar.

En este caso, no se puede apreciar exactamente una relación clara entre estas dos variables, pero si tomamos un rango de días, podamos observar mejor estos fenómenos.

Con ayuda de la función xlim de pyplot, analiza las gráficas del punto anterior para 5 días y trata de explicar si hay o no una clara manifestación de dependencia de Salinidad y Nivel de mar o de Nivel de Mar y Temperatura del agua.

a) Nivel de mar y Salinidad



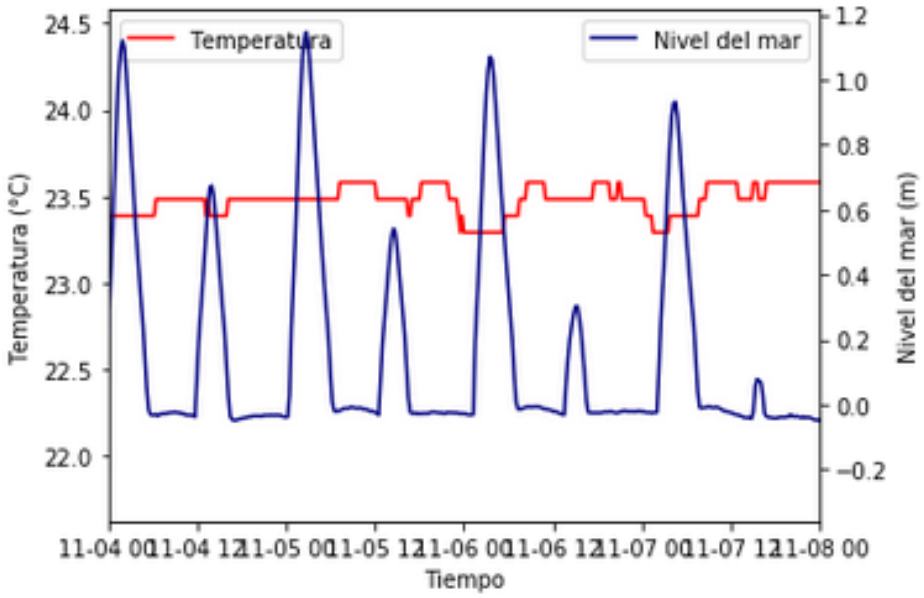
Para realizar esta gráfica se tomo la gráfica anterior correspondiente a estos datos y solamente se modificaron los limites del eje x, siendo este el código: **plt.xlim(("2017-11-4 00:00:00","2017-11-8 00:00:00"))**, en donde se tomo del día 4 de Noviembre hasta el 8.

En este caso podemos observar como al tener niveles bajos de salinidad el nivel del mar tiene picos, lo que significa que si existe una relación entre estas dos variables.

b) Nivel de mar y Temperatura

Al igual que la gráfica anterior, se tomo el mismo intervalo de 4 de noviembre a 8 de noviembre, tomando un intervalo de 5 días. Solamente se tomo la gráfica anterior y se modificaron los limites.

En este intervalo no se puede observar una relación exacta, pero si podemos notar que en algunos de los picos del Nivel del Mar, la Temperatura baja un poco. Por tanto la relación no es tan clara, pero si están relacionadas en alguna manera estas dos variables.



3 Conclusiones

Al terminar esta evaluación me di cuenta que el proceso de limpieza puede facilitarse mas al combinarse tanto las herramientas de pandas con Python y emacs, observando así los datos con ambos programas para ver de que manera es mas eficiente limpiarlos.

Al realizar esta evaluación, también me di cuenta de que aun hay muchas cosas mas por explorar en muchas de las librerías que utilizamos, sobre todo en matplotlib y seaborn, donde las graficas que se pueden realizar con ellas y la información que podemos concluir de ellas nos puede ayudar mucho en nuestra carrera.

Por último, me pareció muy interesante el tema que se trato en esta evaluación y la información que se encontró mediante el uso de las gráficas solicitadas como lo es la correlación entre distintas variables.