Primera Evaluación

Ramses Pacheco Ortiz

March 9, 2018

1 Preparación de los datos

Primeramente, se nos presentan dos archivos a descargar,los descargamos y con ayuda de los comandos de Linux y de Emacs los modificamos para que las cantidad de los datos de los dos archivos sean las mismas, al final los dos archivos se quedaron con 2216 datos.

2 Lectura de datos en jupyter

Ya que tenemos preparados nuestros datos,proseguimos a abrir una terminal y el jupyter notebook, para posterioremente crear una nueva terminal de python 3 y en ésta se descargaron las librerias de panda,numpy,pyplot de matplotlib y la biblioteca de datetime,depsués le dimos estructura a nuestros datos con el dataframe,confirmando con anterioridad los tipos de datos que teniamos.A continuación se muestra una imagen de lo mencionado:

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from datetime import datetime
```

In [2]:
```
df1= pd.read_csv('sargento-270218.csv', header=None, skiprows=1, names=['#','Datetime','AbsPres','Tem
p','waterlevel'])
df2= pd.read_csv('sargentosal-270218.csv', header=None,skiprows=1, names=['#','Datetime','CondHighRng
','Temp','SpecificConductance','salinity'])
```

```
In [4]: df1.head()
```

Out[4]:

	#	Datetime	AbsPres	Temp	waterlevel
0	1	02/04/2018 09:45:00	108.068	17.094	0.060
1	2	02/04/2018 10:00:00	107.815	16.903	0.035
2	3	02/04/2018 10:15:00	107.791	16.903	0.032
3	4	02/04/2018 10:30:00	107.791	16.903	0.032
4	5	02/04/2018 10:45:00	107.791	16.903	0.032

```
In [5]: df2.head()
```

Out[5]:

	#	Datetime	CondHighRng	Temp	SpecificConductance	salinity
0	1	02/04/2018 09:30:00	50782.6	17.53	59295.4	39.6728
1	2	02/04/2018 09:45:00	50617.2	17.33	59362.4	39.7235
2	3	02/04/2018 10:00:00	50646.3	17.23	59526.8	39.8482
3	4	02/04/2018 10:15:00	50772.9	17.17	59753.7	40.0203
4	5	02/04/2018 10:30:00	51007.8	17.14	60068.6	40.2595

```
In [8]: df1=pd.DataFrame(df1)
```

```
In [9]: df2=pd.DataFrame(df2)
```

3 Biblioteca de Seaborn y boxplot

Ahora que ya tenemos las bibliotecas y los datos estructurados, proseguiremos a realizar las gráficas de caja,para esto primero convertimos la cadena de carácteres ”datetime” en una variable temporal ”Ndate” para ambos archivos,a continuación se muestra una gráfica de estos pasos:

```
In [10]: df1['Ndate'] = pd.to_datetime(df1['Datetime'], format='%m/%d/%Y %H:%M:%S')
         df1['month'] = df1['Ndate'].dt.month
         df1.head()
```

Out[10]:

	#	Datetime	AbsPres	Temp	waterlevel	Ndate	month
0	1	02/04/2018 09:45:00	108.068	17.094	0.060	2018-02-04 09:45:00	2
1	2	02/04/2018 10:00:00	107.815	16.903	0.035	2018-02-04 10:00:00	2
2	3	02/04/2018 10:15:00	107.791	16.903	0.032	2018-02-04 10:15:00	2
3	4	02/04/2018 10:30:00	107.791	16.903	0.032	2018-02-04 10:30:00	2
4	5	02/04/2018 10:45:00	107.791	16.903	0.032	2018-02-04 10:45:00	2

In [11]:
```
df2['Ndate'] = pd.to_datetime(df2['Datetime'], format='%m/%d/%Y %H:%M:%S')
df2['month'] = df2['Ndate'].dt.month
df2.head()
```

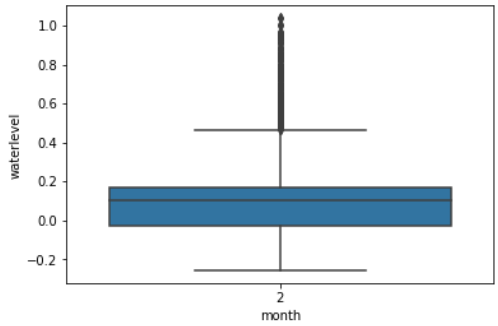
Out[11]:

	#	Datetime	CondHighRng	Temp	SpecificConductance	salinity	Ndate	month
0	1	02/04/2018 09:30:00	50782.6	17.53	59295.4	39.6728	2018-02-04 09:30:00	2
1	2	02/04/2018 09:45:00	50617.2	17.33	59362.4	39.7235	2018-02-04 09:45:00	2
2	3	02/04/2018 10:00:00	50646.3	17.23	59526.8	39.8482	2018-02-04 10:00:00	2
3	4	02/04/2018 10:15:00	50772.9	17.17	59753.7	40.0203	2018-02-04 10:15:00	2
4	5	02/04/2018 10:30:00	51007.8	17.14	60068.6	40.2595	2018-02-04 10:30:00	2

Acontinuación abrimos la biblioteca seaborn y matplotlib para realizar la siguiente gráfica de la variabilidad de los datos de febrero:

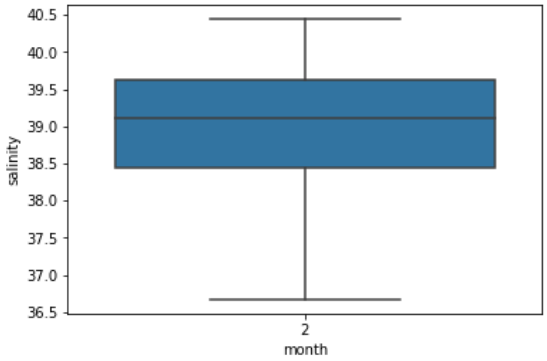
a) Nivel de mar (metros)

In [12]:
```
import seaborn as sns
import matplotlib.pyplot as plt
ax = sns.boxplot(x="month", y="waterlevel", data=df1)
plt.show()
```



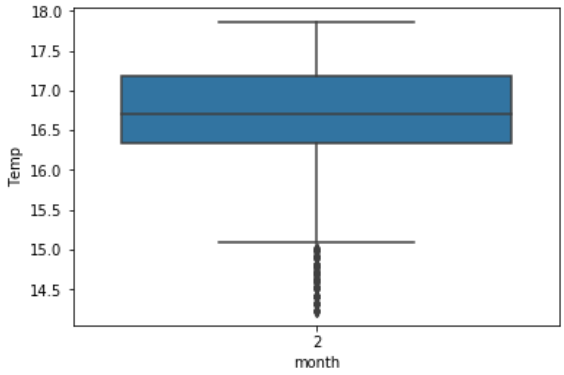
b) Salinidad (Partes por mil - ppt)

```
In [13]: bx = sns.boxplot(x="month", y="salinity", data=df2)
         plt.show()
```



c) Temperatura de Agua (ºC)

```
In [14]: cx=sns.boxplot(x="month", y="Temp", data=df1)
         plt.show()
```



4 Regresión lineal con las distribuciones marginales

En este punto de la actividad volvimos a utilizar la biblioteca seaborn y también exploramos si hay una correlación de Pearson entre cada pareja de variables, utilizamos el comando concat(concatenar):

- Nivel de mar-Salinidad

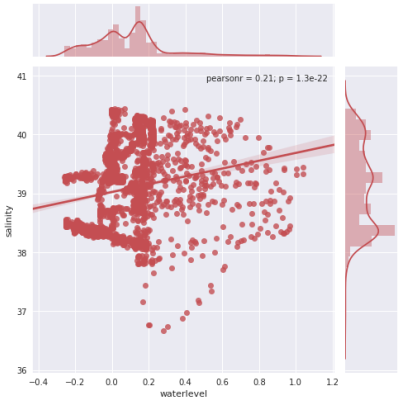
  Como estas dos variables están en diferentes archivos, tuvimos que hacer otra estructura de datos (dateframe) para poder llevar acabo ésta gráfica;a continuación se muestra el código y la gráfica resultante.

In [18]:
```
df3=pd.concat([df1['waterlevel'], df2['salinity']],axis=1)
df3.head(5)
```

Out[18]:

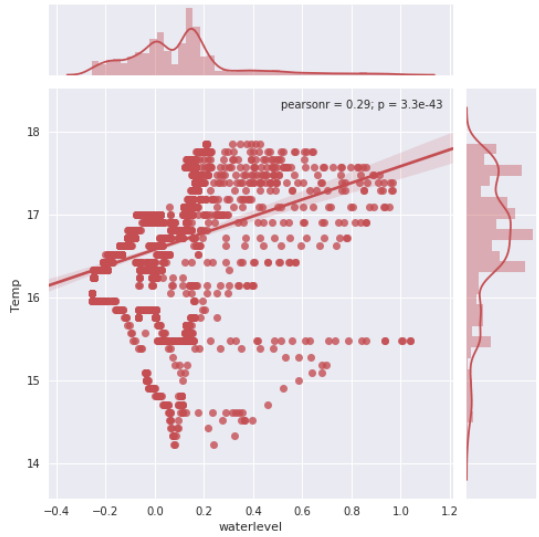
	waterlevel	salinity
0	0.060	39.6728
1	0.035	39.7235
2	0.032	39.8482
3	0.032	40.0203
4	0.032	40.2595

In [19]:
```
import seaborn as sns
sns.set(style="darkgrid", color_codes=True)

g = sns.jointplot("waterlevel", "salinity", data=df3, kind="reg",
                  color="r", size=7)
df3=pd.concat([df1['waterlevel'], df2['salinity']],axis=1)
plt.show(g)
```



- Nivel de mar-Temperatura del agua En éste caso las dos variables están en el mismos archivo, no se necesito hacerles cambio ya que anteriormente ya los habiamos estructurado:

In [15]:
```
import seaborn as sns
sns.set(style="darkgrid", color_codes=True)

g = sns.jointplot("waterlevel", "Temp", data=df1, kind="reg",
                  color="r", size=7)
plt.show(g)
```



- Salinidad-Temperatura del agua

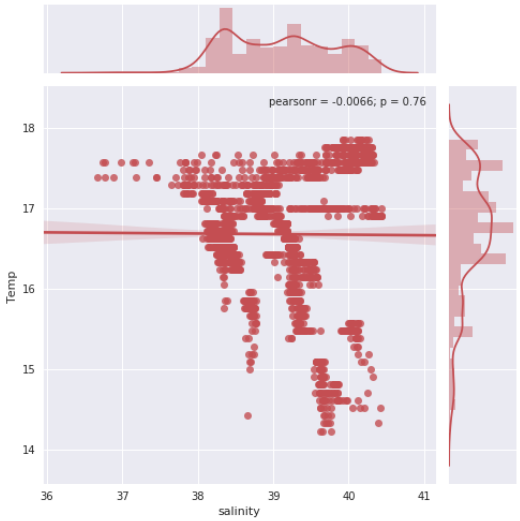
  Igualmente que en el primer inciso los datos no se encuentran en la misma carpeta y realizamos lo mismo(dataframe).

In [16]:
```
df4= pd.concat([df2['salinity'], df1['Temp']],axis=1)
df4.head(5)
```

Out[16]:

	salinity	Temp
0	39.6728	17.094
1	39.7235	16.903
2	39.8482	16.903
3	40.0203	16.903
4	40.2595	16.903

```
In [17]: import seaborn as sns
sns.set(style="darkgrid", color_codes=True)

g = sns.jointplot("salinity", "Temp", data=df4, kind="reg",
                  color="r", size=7)
df4= pd.concat([df2['salinity'], df1['Temp']],axis=1)
plt.show(g)
```

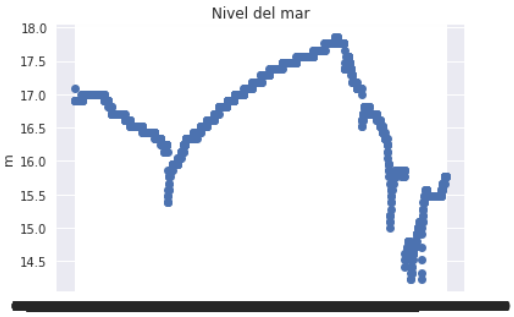


5 Gráficas independientes de las variables

En ésta sección realizamos tres gráficas idependientes de las variables en función del tiempo, acontinuación se mostraran las 3 gráficas con su respectivo código.

- : Nivel del mar

```
In [24]: plt.plot_date(x=df1.Datetime, y=df1['Temp'], fmt="b-")
plt.title("Nivel del mar")
plt.ylabel("m")
plt.grid(True)
plt.show()
```

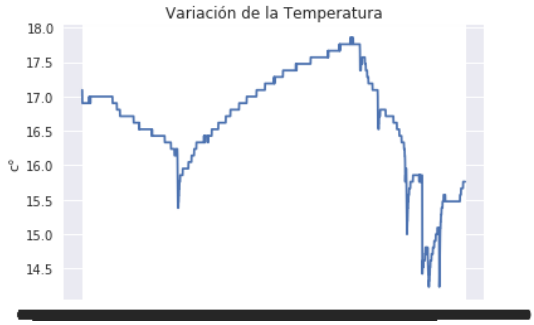


- Salinidad

```
In [25]: plt.plot_date(x=df2.Datetime, y=df2['salinity'], fmt="b-")
         plt.title("Variación de la salinidad")
         plt.ylabel("ppt")
         plt.grid(True)
         plt.show()
```

- Temperatura del agua

```
In [27]: plt.plot_date(x=df2.Datetime, y=df1['Temp'], fmt="b-")
         plt.title("Variación de la Temperatura")
         plt.ylabel("c°")
         plt.grid(True)
         plt.show()
```

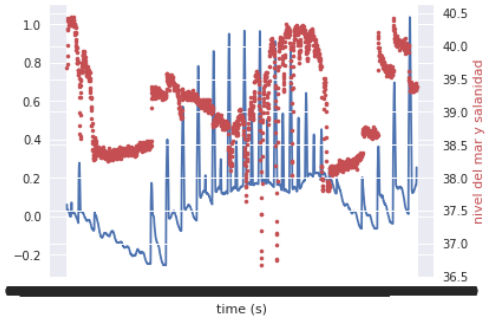


6 Gráficas con doble eje vertical

En este punto tuvimos que buscar en internet un nuevo código para realizar las siguientes gráficas que se mostrarán.

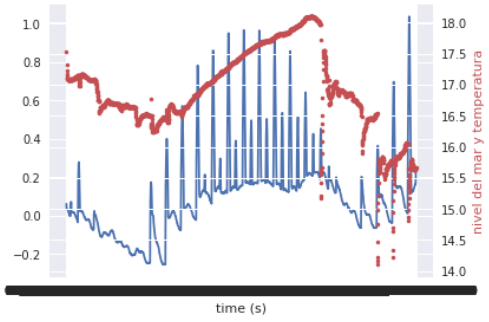
- Nivel de mar y salinidad

In [30]:
```
fig, ax1 = plt.subplots()
a=df1.waterlevel
b=df2.salinity
t=df1.Datetime
ax1.plot(t, a, 'b-')
ax1.set_xlabel('time (s)')

ax2 = ax1.twinx()
ax2.plot(t, b, 'r.')
ax2.set_ylabel('nivel del mar y salanidad', color='r')
fig.tight_layout()

plt.show()
```



- Nivel de mar y temperatura

In [31]:
```
fig, ax1 = plt.subplots()
a=df1.waterlevel
b=df2.Temp
t=df1.Datetime
ax1.plot(t, a, 'b-')
ax1.set_xlabel('time (s)')

ax2 = ax1.twinx()
ax2.plot(t, b, 'r.')
ax2.set_ylabel('nivel del mar y temperatura', color='r')
fig.tight_layout()

plt.show()
```

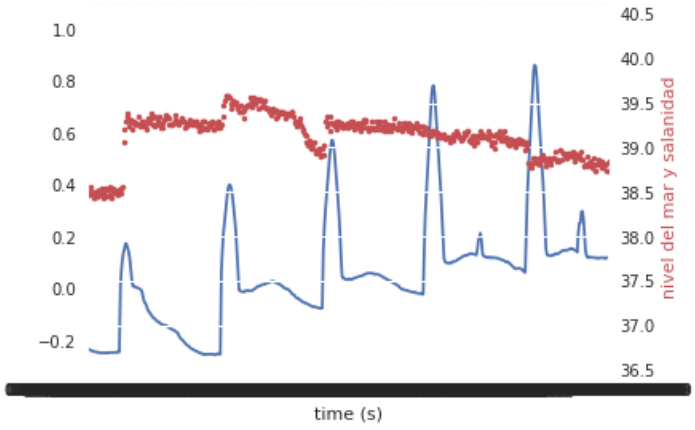


7 Xlim

En este último punto, se probo la funcion o comando xlim para establecer el limite de tiempo,debido a los datos que teniamos, establecimos un limite en donde marcara diferencia en comparación con la cantidad de datos, a continuación se mostrará el código y la gráfica resultante.

- Nivel de mar y salinidad

```
In [39]: fig, ax1 = plt.subplots()
a=df1.waterlevel
b=df2.salinity
t=df1.Datetime
ax1.plot(t, a, 'b-')
ax1.set_xlabel('time (s)')

ax2 = ax1.twinx()
ax2.plot(t, b, 'r.')
ax2.set_ylabel('nivel del mar y salanidad', color='r')
fig.tight_layout()
plt.xlim((500,1000))
plt.show()
```



- Nivel de mar y temperatura

In [40]:

```
fig, ax1 = plt.subplots()
a=df1.waterlevel
b=df2.Temp
t=df1.Datetime
ax1.plot(t, a, 'b-')
ax1.set_xlabel('time (s)')

ax2 = ax1.twinx()
ax2.plot(t, b, 'r.')
ax2.set_ylabel('nivel del mar y temperatura', color='r')
fig.tight_layout()

plt.xlim((500,1000))
plt.show()
```