Linear Regression - Salinity and Temperature

In [1]:

```
#Step1: Importing All the Required Libraries

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing, svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

```
In [2]: #Step2: Reading the Dataset

        df=pd.read_csv(r"C:\Users\sneha\Downloads\bottle.csv.zip")
        df
```

```
C:\Users\sneha\AppData\Local\Temp\ipykernel_12900\2303548971.py:3: DtypeWarning: Columns (47,73) hav
e mixed types. Specify dtype option on import or set low_memory=False.
  df=pd.read_csv(r"C:\Users\sneha\Downloads\bottle.csv.zip")
```

Out[2]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2Sat	...	R_PHAEO	R_PRES
0	1	1	054.0 056.0	19-4903CR-HY-060-0930-05400560-0000A-3	0	10.500	33.4400	NaN	25.64900	NaN	...	NaN	0
1	1	2	054.0 056.0	19-4903CR-HY-060-0930-05400560-0008A-3	8	10.460	33.4400	NaN	25.65600	NaN	...	NaN	8
2	1	3	054.0 056.0	19-4903CR-HY-060-0930-05400560-0010A-7	10	10.460	33.4370	NaN	25.65400	NaN	...	NaN	10
3	1	4	054.0 056.0	19-4903CR-HY-060-0930-05400560-0019A-3	19	10.450	33.4200	NaN	25.64300	NaN	...	NaN	19
4	1	5	054.0 056.0	19-4903CR-HY-060-0930-05400560-0020A-7	20	10.450	33.4210	NaN	25.64300	NaN	...	NaN	20
...	...	...	...	...	...	...	...	...	...	...	...	...	...
864858	34404	864859	093.4 026.4	20-1611SR-MX-310-2239-09340264-0000A-7	0	18.744	33.4083	5.805	23.87055	108.74	...	0.18	0
864859	34404	864860	093.4 026.4	20-1611SR-MX-310-2239-09340264-0002A-3	2	18.744	33.4083	5.805	23.87072	108.74	...	0.18	2
864860	34404	864861	093.4 026.4	20-1611SR-MX-310-2239-09340264-0005A-3	5	18.692	33.4150	5.796	23.88911	108.46	...	0.18	5
864861	34404	864862	093.4 026.4	20-1611SR-MX-310-2239-09340264-0010A-3	10	18.161	33.4062	5.816	24.01426	107.74	...	0.31	10
864862	34404	864863	093.4 026.4	20-1611SR-MX-310-2239-09340264-0015A-3	15	17.533	33.3880	5.774	24.15297	105.66	...	0.61	15

864863 rows × 74 columns

In [3]:
```
df=df[['Salnty','T_degC']]
df.columns=['Sal','Temp']
```

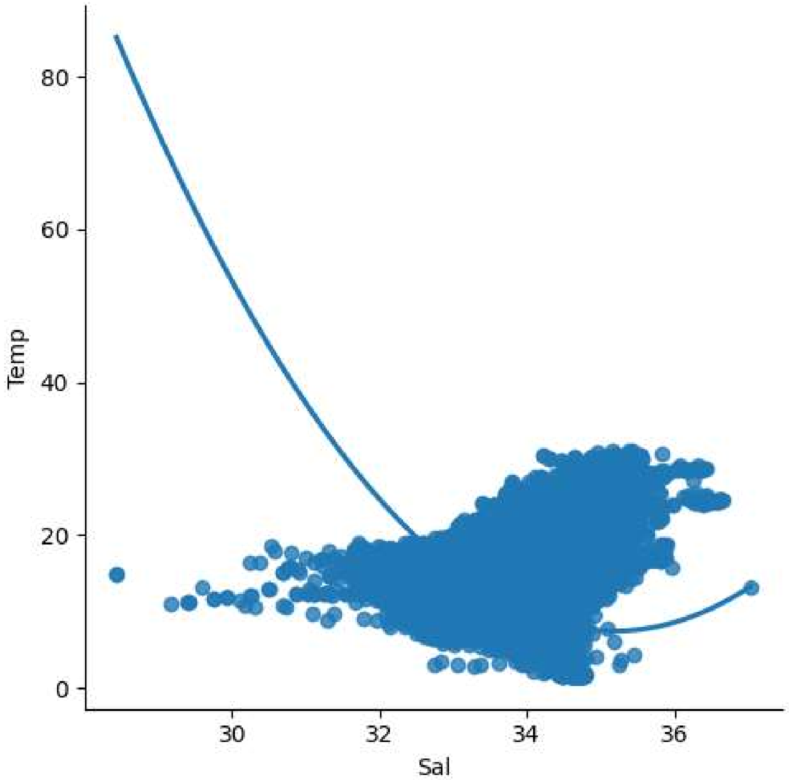
In [4]:
```
df.head(10)
```

Out[4]:

	Sal	Temp
0	33.440	10.50
1	33.440	10.46
2	33.437	10.46
3	33.420	10.45
4	33.421	10.45
5	33.431	10.45
6	33.440	10.45
7	33.424	10.24
8	33.420	10.06
9	33.494	9.86

In [5]:
```
#step3: Exploring  the Data  Scatter . Plotting the data scatter

sns.lmplot(x ="Sal", y="Temp", data =df, order = 2, ci = None)
```

Out[5]:
```
<seaborn.axisgrid.FacetGrid at 0x1ee5bb30ca0>
```



In [6]:

```
#Step4: Data Cleaning- Elimminating NaN or missing input numbers

df.fillna(method= 'ffill', inplace = True)
```

```
C:\Users\sneha\AppData\Local\Temp\ipykernel_12900\1577053232.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexi
ng.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/inde
xing.html#returning-a-view-versus-a-copy)
  df.fillna(method= 'ffill', inplace = True)
```

In [7]:

```
#Step5: Training our Model

X = np.array(df['Sal']).reshape(-1, 1)
y = np.array(df['Temp']).reshape(-1, 1)

#Separating the data into independent and dependent variables and convert
#Now each dataframe contains only one column
```

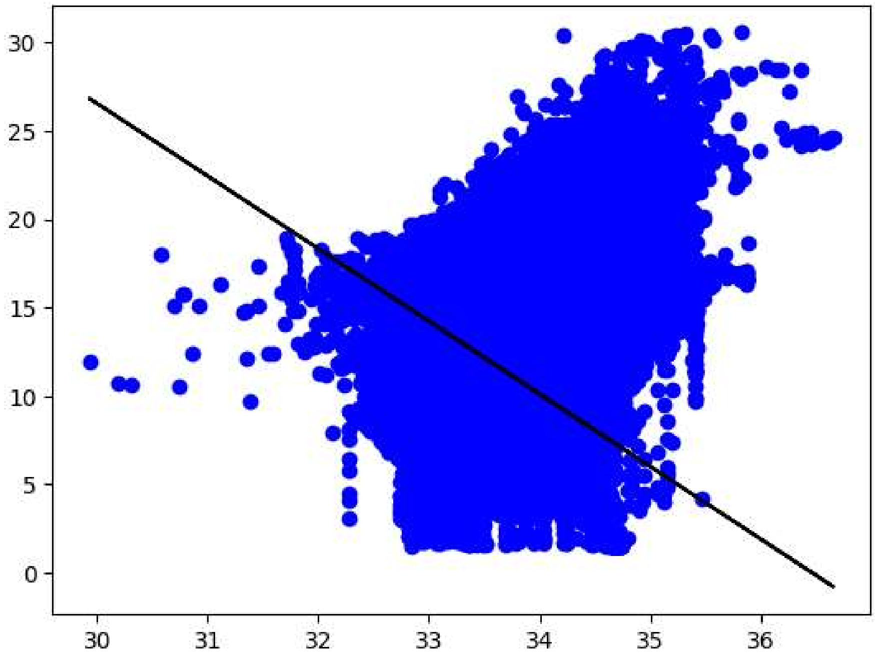
In [8]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25)
#Splitting the data into training and testing data
regr = LinearRegression()
regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

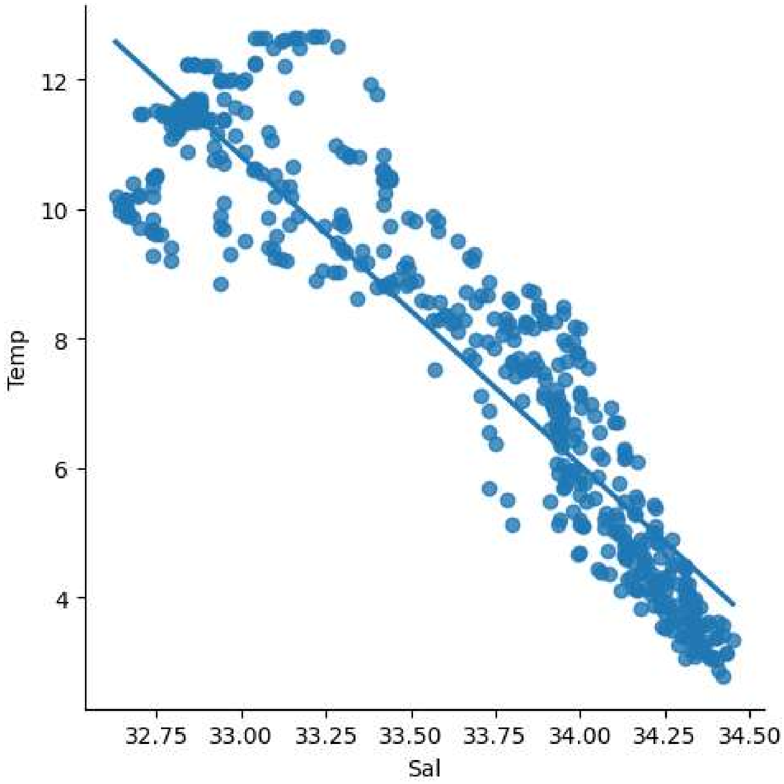
```
0.20408467365361427
```

In [9]:

```
#Step6: Exploring our Results
y_pred = regr.predict(X_test)
plt.scatter(X_test, y_test, color = 'b')
plt.plot(X_test, y_pred,color = 'k')
plt.show()
```



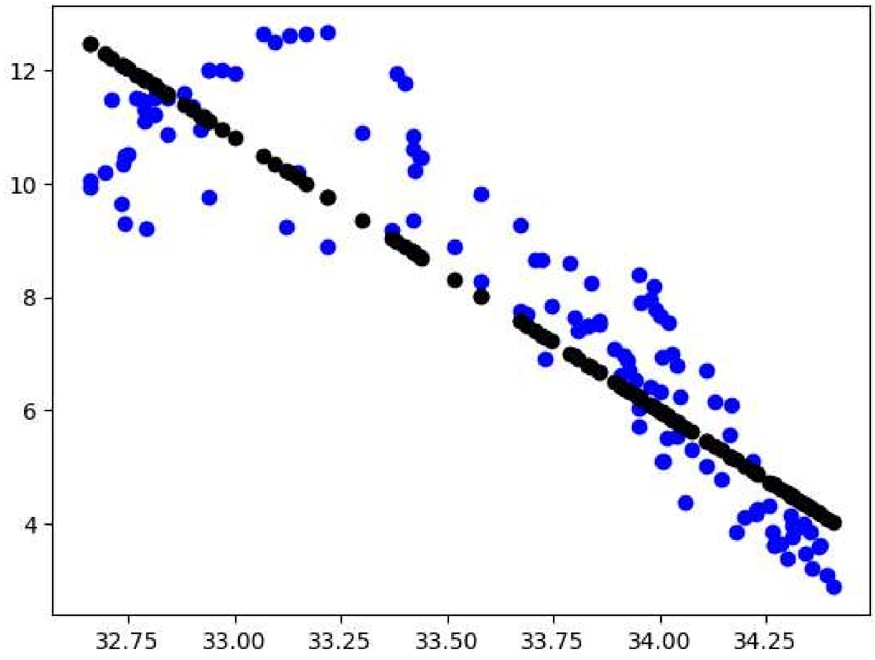
```
In [10]: #Step7: Working with a Smaller dataset
         df500 = df[:][:500]

         #Selecting the 1st 500 rowss of the data
         sns.lmplot(x= "Sal", y="Temp", data= df500, order = 1, ci = None)

Out[10]: <seaborn.axisgrid.FacetGrid at 0x1ee08c79f00>
```



In [11]:

```
df500.fillna(method = 'ffill', inplace = True)
X = np.array(df500['Sal']).reshape(-1, 1)
y = np.array(df500['Temp']).reshape(-1, 1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25)
regr = LinearRegression()
regr.fit(X_train, y_train)
print("Regression:",regr.score(X_test, y_test))
y_pred=regr.predict(X_test)
plt.scatter(X_test,y_test,color='b')
plt.scatter(X_test,y_pred,color='k')
plt.show()
```

```
Regression: 0.8118937504571804
```



In [12]:

```
#Step8: Evaluation of model
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
#train the model
model = LinearRegression()
model.fit(X_train, y_train)
# Evaluate the model on the test set
y_pred = model.predict(X_test)

r2 = r2_score(y_test, y_pred)

print("R2 score:",r2)
```

```
R2 score: 0.8118937504571804
```

In [ ]:

```
#Step-9:Conclusion:

Dataset we have taken is poor for Linear Model,but with the smaller data works well with Linear Model
```

In [ ]: