In [1]:
```
#step 1: 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]:
```
df = pd.read_csv(r"C:\Users\Sudheer\Downloads\bottle.csv.zip",low_memory=False
```

In [3]:
```
df
```

Out[3]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2S
0	1	1	054.0 056.0	19-4903CR-HY-060-0930-05400560-0000A-3	0	10.500	33.4400	NaN	25.64900	Na
1	1	2	054.0 056.0	19-4903CR-HY-060-0930-05400560-0008A-3	8	10.460	33.4400	NaN	25.65600	Na
2	1	3	054.0 056.0	19-4903CR-HY-060-0930-05400560-0010A-7	10	10.460	33.4370	NaN	25.65400	Na
3	1	4	054.0 056.0	19-4903CR-HY-060-0930-05400560-0019A-3	19	10.450	33.4200	NaN	25.64300	Na
4	1	5	054.0 056.0	19-4903CR-HY-060-0930-05400560-0020A-7	20	10.450	33.4210	NaN	25.64300	Na
...	...	...	...	...	...	...	...	...	...	
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.
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.
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.
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.

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2S
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.

864863 rows × 74 columns

In [4]:
```
df=df[['Salnty','T_degC']]
df.columns=['Sal','Temp']
```

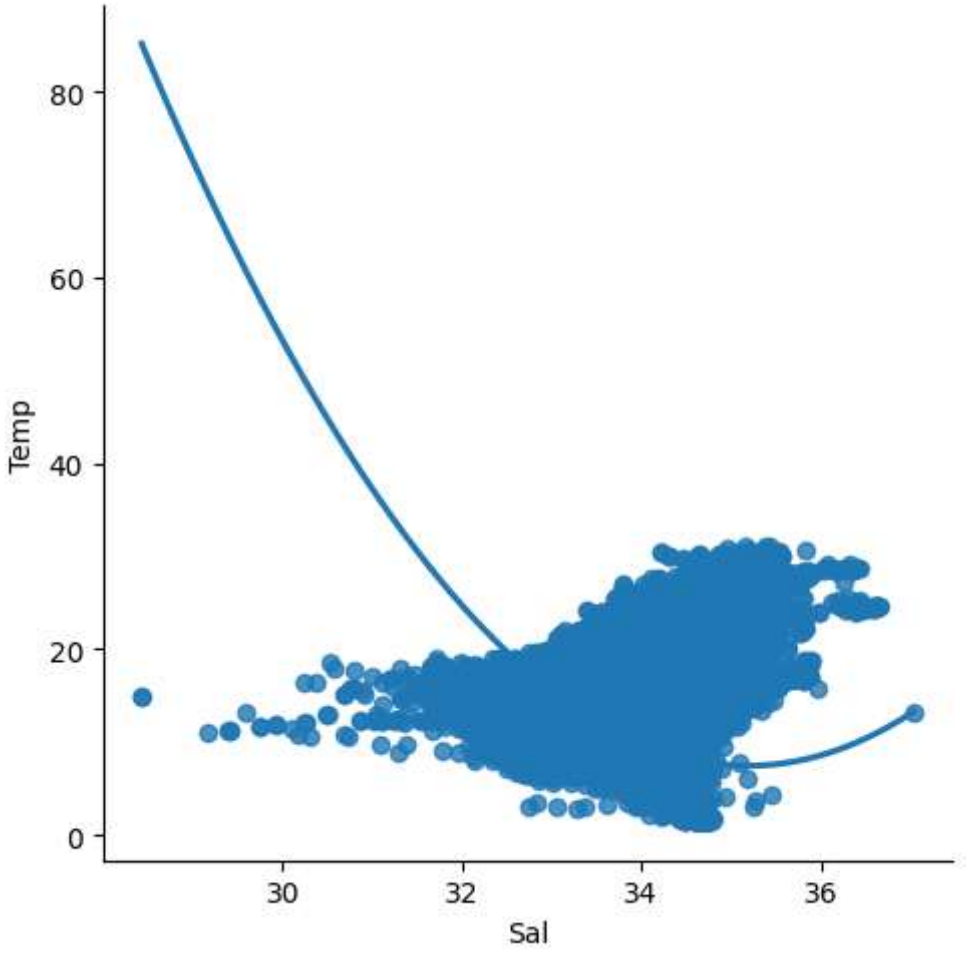
In [5]:
```
df.head(10)

# displaying only the 1st rows along with the
```

Out[5]:

	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 [6]:
```
sns.lmplot(x="Sal",y="Temp",data=df,order=2,ci=None)
```

Out[6]: <seaborn.axisgrid.FacetGrid at 0x206090d6b60>



In [7]:
```
df.describe()
```

Out[7]:

	Sal	Temp
count	817509.000000	853900.000000
mean	33.840350	10.799677
std	0.461843	4.243825
min	28.431000	1.440000
25%	33.488000	7.680000
50%	33.863000	10.060000
75%	34.196900	13.880000
max	37.034000	31.140000

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 864863 entries, 0 to 864862
Data columns (total 2 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Sal     817509 non-null  float64
 1   Temp    853900 non-null  float64
dtypes: float64(2)
memory usage: 13.2 MB
```

In [9]:
```
#step 4: Data cleaning - Eliminating NaN or missing input numbers

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

```
C:\Users\Sudheer\AppData\Local\Temp\ipykernel_13532\2434860272.py:3: SettingW
ithCopyWarning:
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/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  df.fillna(method='ffill',inplace=True)
```

In [10]:
```
x = np.array(df['Sal']).reshape(-1, 1)

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

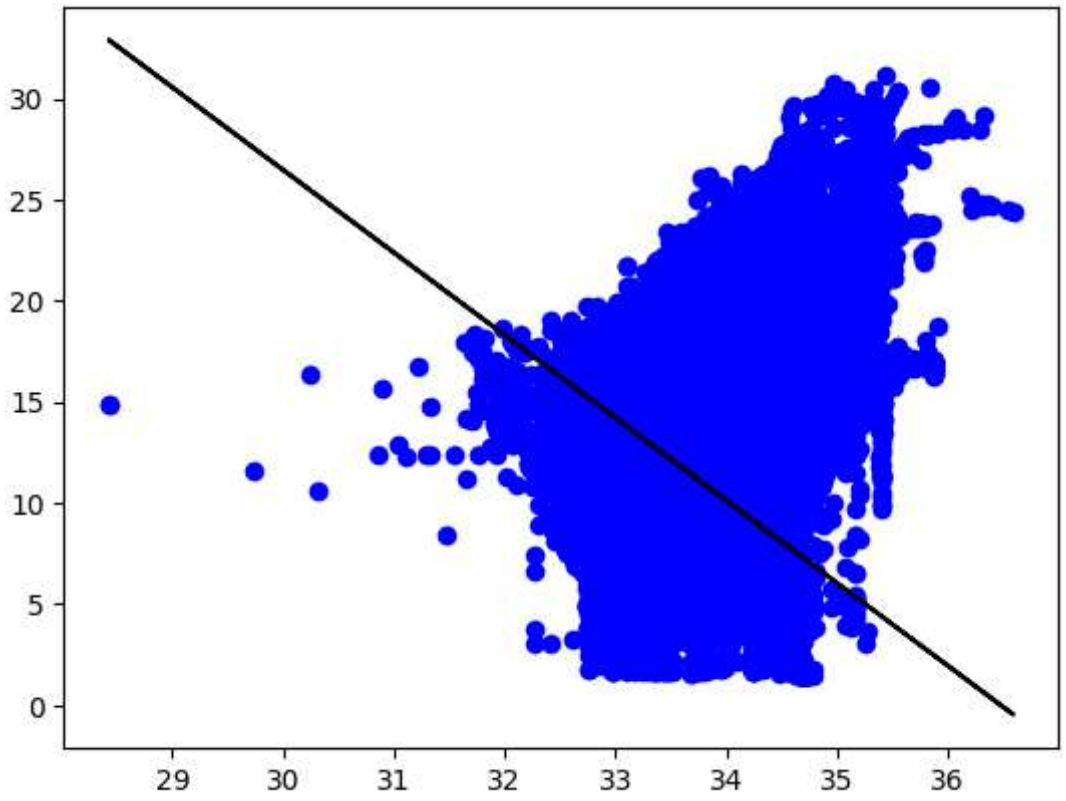
In [11]:
```
df.dropna(inplace = True)

# Dropping any rows with nan values
```

```
C:\Users\Sudheer\AppData\Local\Temp\ipykernel_13532\3158782430.py:1: SettingW
ithCopyWarning:
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/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  df.dropna(inplace = True)
```

In [12]:
```
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(regr.score(x_test,y_test))
```

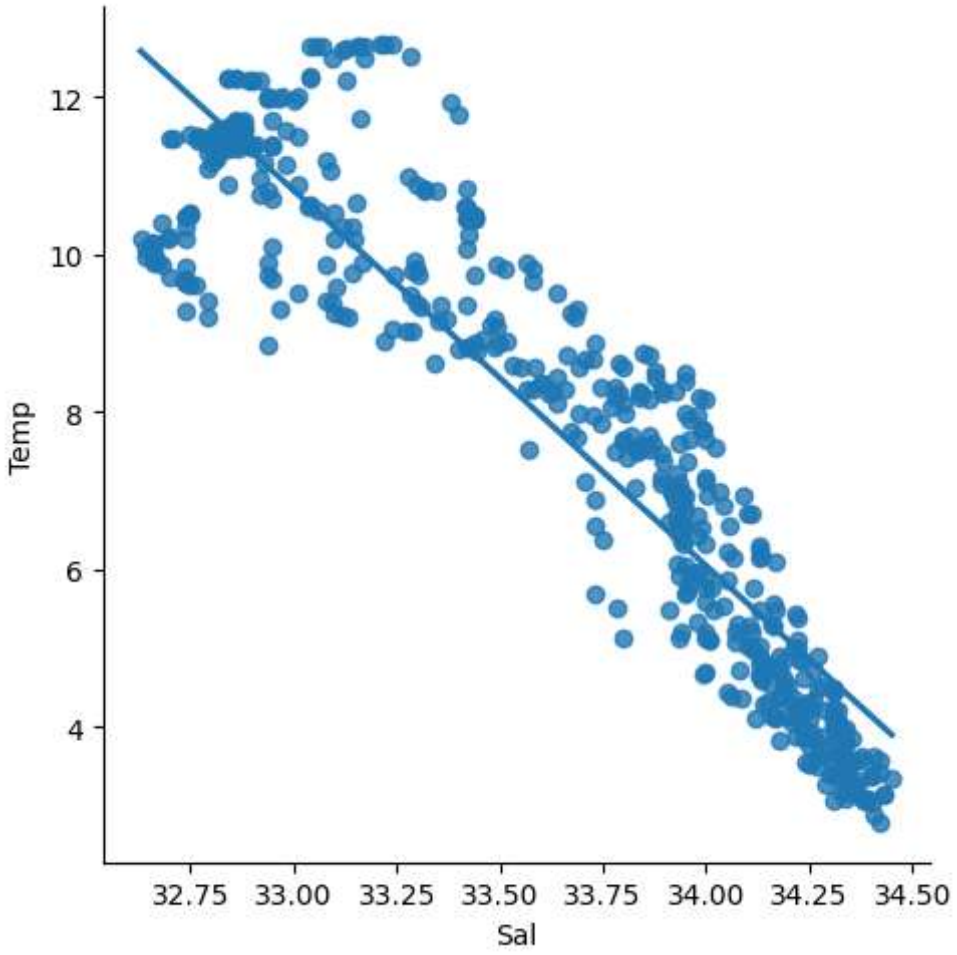
```
0.20830084385093084
```

In [13]:
```
# step6:

y_pred = regr.predict(x_test)

plt.scatter(x_test, y_test, color ='b')

plt.plot(x_test, y_pred,color = 'k')

plt.show()
```



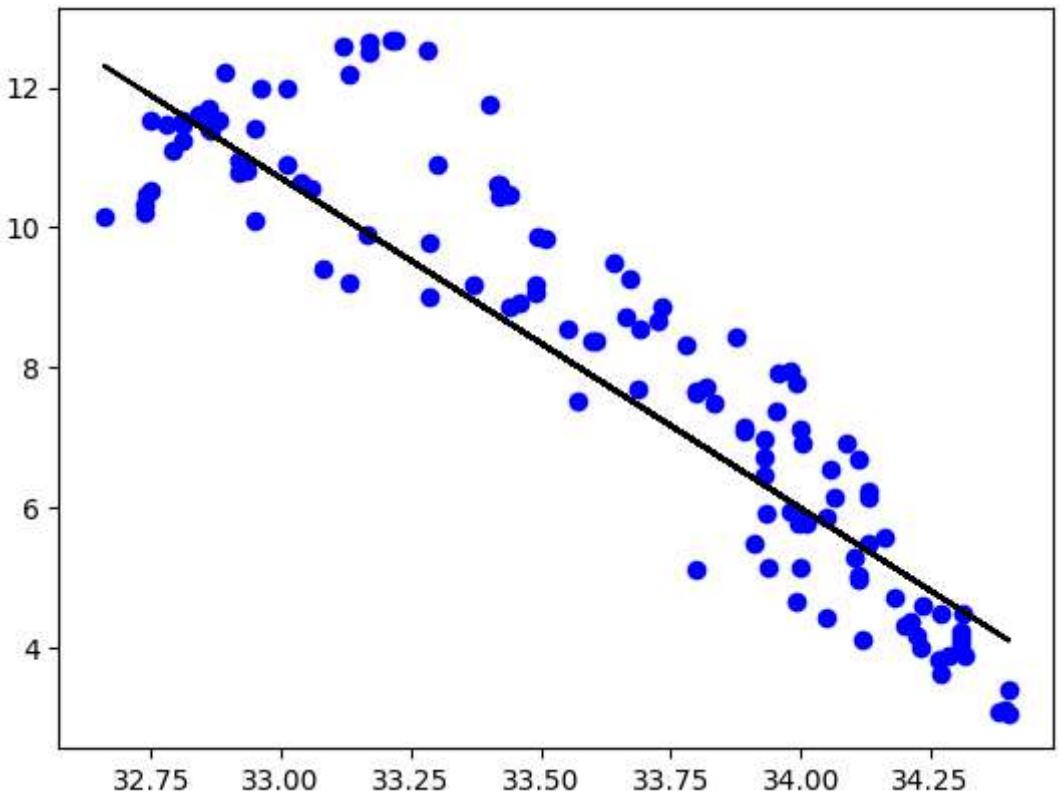
In [14]:
```
# step7: working with smaller dataset

df500 = df[:][:500]

# selecting the 1st 500 rows of the data

sns.lmplot(x = "Sal",y = "Temp",data = df500, order = 1, ci = None)
```

Out[14]: <seaborn.axisgrid.FacetGrid at 0x20609f6e860>



In [15]:
```
df500.fillna(method ='ffill',inplace = True)

x = np.array(df500['Sal']).reshape(-1, 1)

y = np.array(df500['Temp']).reshape(-1, 1)

df500.dropna(inplace=True)

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.plot(x_test, y_pred, color = 'k')

plt.show()
```

```
Regression:  0.8243914870413185
```



In [16]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
model = LinearRegression()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
r2 = r2_score(y_test,y_pred)
print("R2 score:",r2)
```

```
R2 score: 0.8243914870413185
```

In [ ]:

In [ ]: