In [3]:
```
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 [6]:
```
df=pd.read_csv(r"C:\Users\P. VIJAY KUMAR\Downloads\bottle.csv.zip")
df
```

```
C:\Users\P. VIJAY KUMAR\AppData\Local\Temp\ipykernel_32188\2771977914.py:1: DtypeWarning: Columns (47,73) have mixed types. Specify dtype option on import or set low_memory=False.
  df=pd.read_csv(r"C:\Users\P. VIJAY KUMAR\Downloads\bottle.csv.zip")
```

Out[6]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2
0	1	1	054.0 056.0	19-4903CR-HY-060-0930-05400560-0000A-3	0	10.500	33.4400	NaN	25.64900	N
1	1	2	054.0 056.0	19-4903CR-HY-060-0930-05400560-0008A-3	8	10.460	33.4400	NaN	25.65600	N
2	1	3	054.0 056.0	19-4903CR-HY-060-0930-05400560-0010A-7	10	10.460	33.4370	NaN	25.65400	N
3	1	4	054.0 056.0	19-4903CR-HY-060-0930-05400560-0019A-3	19	10.450	33.4200	NaN	25.64300	N
4	1	5	054.0 056.0	19-4903CR-HY-060-0930-05400560-0020A-7	20	10.450	33.4210	NaN	25.64300	N
...	...	...	...	...	...	...	...	...	...	
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

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2
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
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 [7]:
```
df=df[['Salnty','T_degC']]
df.columns=['sal','temp']
```

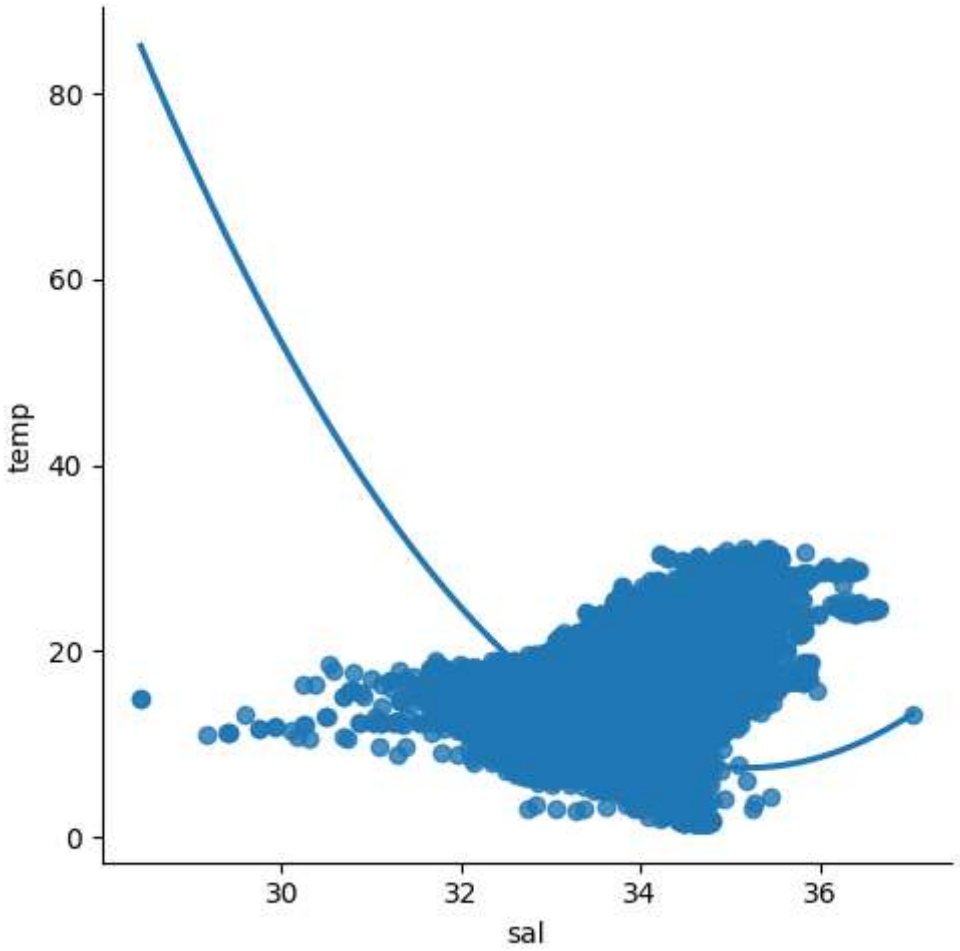
In [14]:
```
df.head(10)
```

Out[14]:

	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 [8]:
```
sns.lmplot(x="sal",y="temp",data=df,order=2,ci=None)
```

Out[8]:
```
<seaborn.axisgrid.FacetGrid at 0x1ad06c80760>
```



In [9]:
```
df.describe()
```

Out[9]:

	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 [10]:
```
df.isna().any()
```

Out[10]:
```
sal     True
temp    True
dtype: bool
```

In [11]:
```
df
```

Out[11]:

	sal	temp
0	33.4400	10.500
1	33.4400	10.460
2	33.4370	10.460
3	33.4200	10.450
4	33.4210	10.450
...	...	...
864858	33.4083	18.744
864859	33.4083	18.744
864860	33.4150	18.692
864861	33.4062	18.161
864862	33.3880	17.533

864863 rows × 2 columns

In [12]:
```
column_to_fill=['sal','temp']
for column in column_to_fill:
    column_mean=df[column].mean()
    df[column].fillna(column_mean,inplace=True)
```

```
C:\Users\P. VIJAY KUMAR\AppData\Local\Temp\ipykernel_32188\1608319408.py:4: Set
tingWithCopyWarning:
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/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  df[column].fillna(column_mean,inplace=True)
```

In [13]:
```
df
```

Out[13]:

	sal	temp
0	33.4400	10.500
1	33.4400	10.460
2	33.4370	10.460
3	33.4200	10.450
4	33.4210	10.450
...	...	...
864858	33.4083	18.744
864859	33.4083	18.744
864860	33.4150	18.692
864861	33.4062	18.161
864862	33.3880	17.533

864863 rows × 2 columns

In [14]:
```
df.isna().any()
```

Out[14]:
```
sal     False
temp    False
dtype: bool
```

In [15]:
```
x=np.array(df['sal']).reshape(-1,1)
y=np.array(df['temp']).reshape(-1,1)
```

In [16]:
```
df.dropna()
```

Out[16]:

	sal	temp
0	33.4400	10.500
1	33.4400	10.460
2	33.4370	10.460
3	33.4200	10.450
4	33.4210	10.450
...	...	...
864858	33.4083	18.744
864859	33.4083	18.744
864860	33.4150	18.692
864861	33.4062	18.161
864862	33.3880	17.533

864863 rows × 2 columns

In [17]:
```
X_train,X_test,Y_train,Y_test=train_test_split(x,y,test_size=0.25)
```

In [18]:
```
regr=LinearRegression()
```

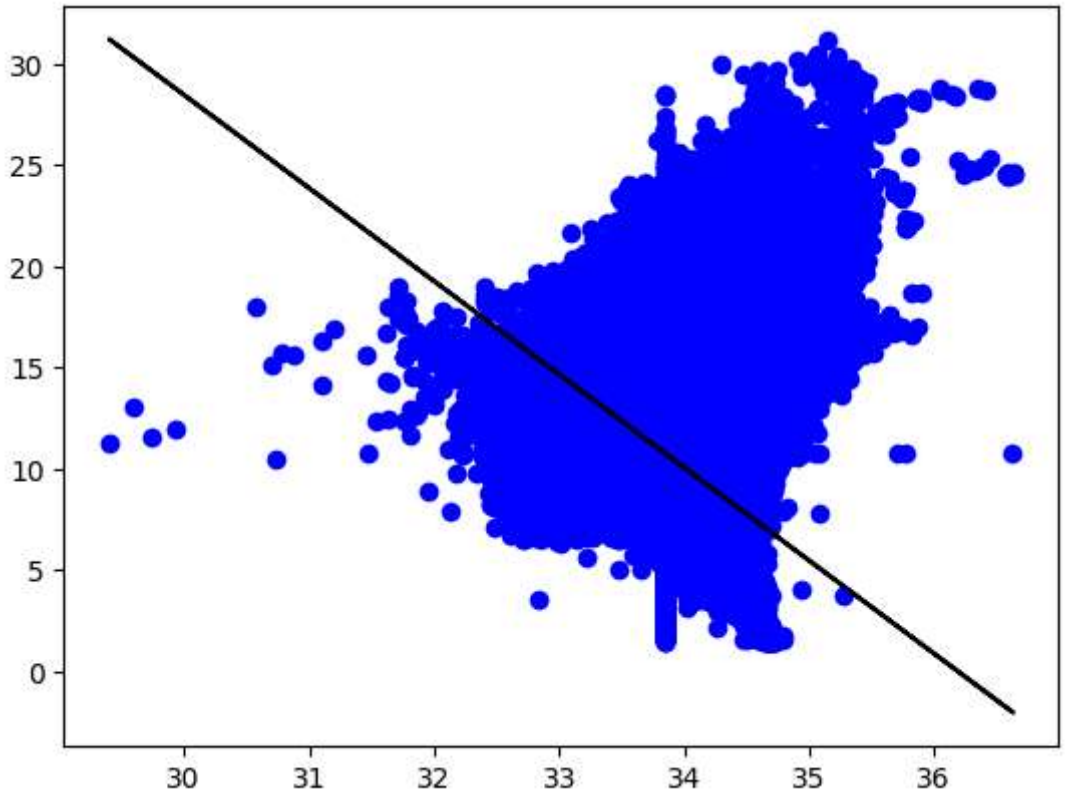
In [19]:
```
regr.fit(X_train,Y_train)
```

Out[19]:
```
LinearRegression()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [20]:
```
print(regr.score(X_test,Y_test))
```

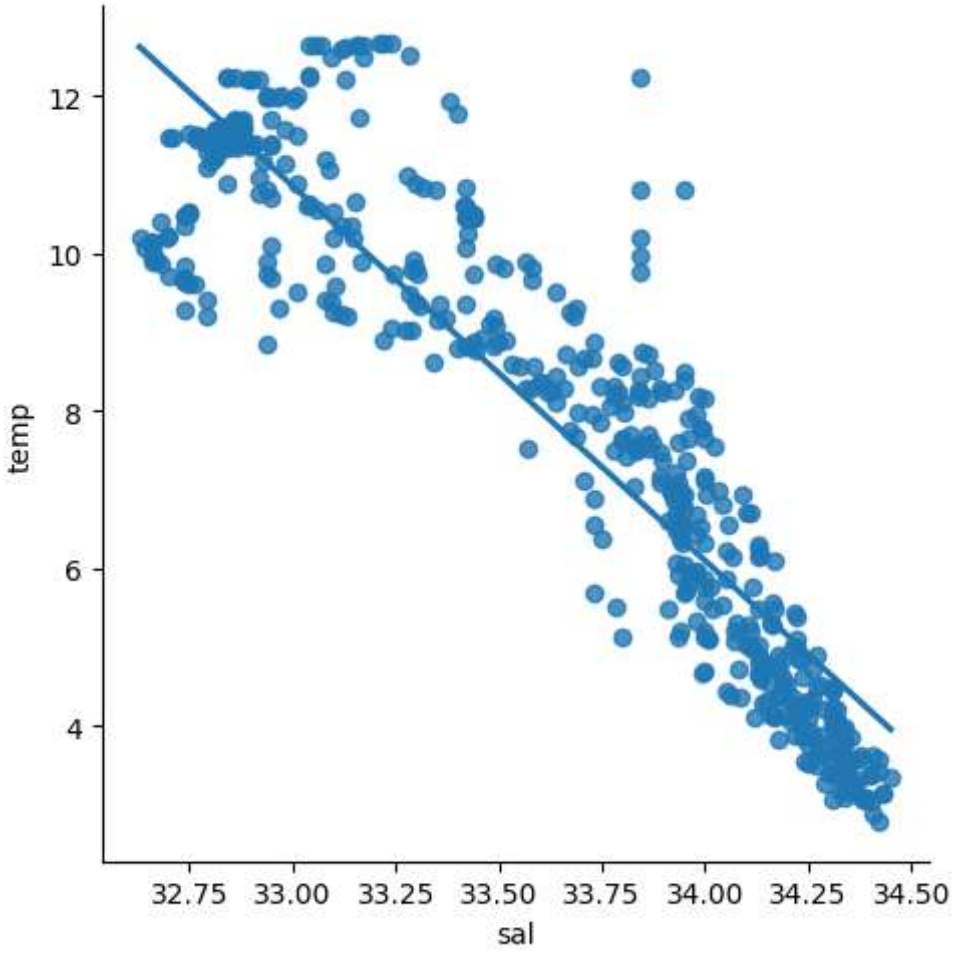
```
0.24173138208189238
```

In [22]:
```
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 [49]:
```
df2=df[:][:500]
sns.lmplot(x="sal",y="temp",data=df2,order=1,ci=None)
```

Out[49]: <seaborn.axisgrid.FacetGrid at 0x1ad312cb640>



In [24]:
```
df2.fillna(method='bfill',inplace=True)
```

In [26]:
```
x=np.array(df2['sal']).reshape(-1,1)
y=np.array(df2['temp']).reshape(-1,1)
df2.dropna()
```

Out[26]:

	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
...	...	...
495	34.269	4.90
496	34.310	4.50
497	34.311	4.48
498	34.319	4.21
499	34.329	3.95

500 rows × 2 columns

In [27]:
```
df2.isna().any()
```

Out[27]:
```
sal     False
temp    False
dtype: bool
```

In [28]:
```
X_train,X_test,Y_train,Y_test=train_test_split(x,y,test_size=0.25)
```

In [29]:
```
regr=LinearRegression()
```

In [30]:
```
regr.fit(X_train,Y_train)
```

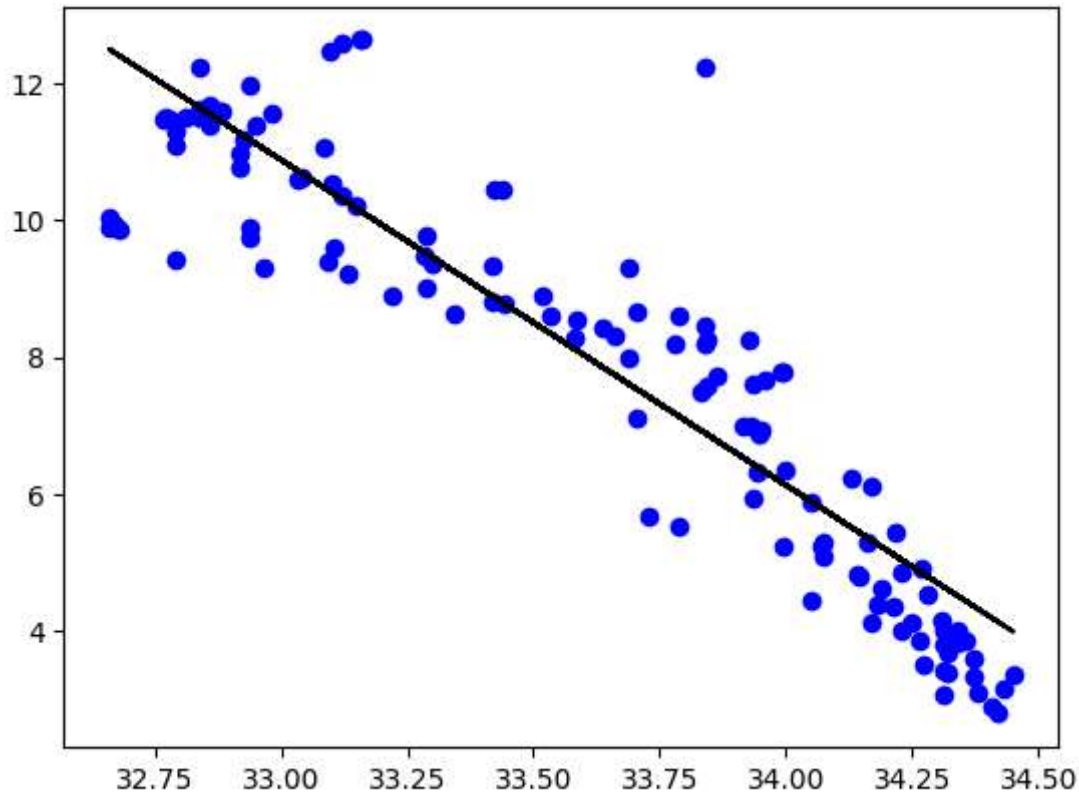
Out[30]:
```
LinearRegression()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [31]:
```
print("Regression:",regr.score(X_test,Y_test))
```

```
Regression: 0.8410043229494959
```

In [32]:
```
Y_pred=regr.predict(X_test)
```

In [38]:
```
plt.scatter(X_test,Y_test,color="b")
plt.plot(X_test,Y_pred,color="k")
plt.show()
```



In [40]:
```
from sklearn.linear_model import LinearRegression
```

In [41]:
```
from sklearn.metrics import r2_score
model=LinearRegression()
```

In [42]:
```
model.fit(X_train,Y_train)
```

Out[42]:
```
LinearRegression()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [43]:
```
Y_pred=model.predict(X_test)
```

In [44]:
```
r2=r2_score(Y_test,Y_pred)
```

In [45]:
```
print("R2 Score:",r2)
```

```
R2 Score: 0.8410043229494959
```

In [ ]:
```

```