Salinity Temperature

In [2]:
```
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 [3]:
```
df=pd.read_csv(r"C:\Users\manis\Downloads\bottle.csv.zip")
df
```

```
C:\Users\manis\AppData\Local\Temp\ipykernel_5572\726471867.py:1: DtypeWarning: Co
lumns (47,73) have mixed types. Specify dtype option on import or set low_memory=
False.
  df=pd.read_csv(r"C:\Users\manis\Downloads\bottle.csv.zip")
```

Out[3]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STh
0	1	1	054.0 056.0	19-4903CR-HY-060-0930-05400560-0000A-3	0	10.500	33.4400	NaN	25.649
1	1	2	054.0 056.0	19-4903CR-HY-060-0930-05400560-0008A-3	8	10.460	33.4400	NaN	25.656
2	1	3	054.0 056.0	19-4903CR-HY-060-0930-05400560-0010A-7	10	10.460	33.4370	NaN	25.654
3	1	4	054.0 056.0	19-4903CR-HY-060-0930-05400560-0019A-3	19	10.450	33.4200	NaN	25.643
4	1	5	054.0 056.0	19-4903CR-HY-060-0930-05400560-0020A-7	20	10.450	33.4210	NaN	25.643
...	...	...	...	...	...	...	...	...	
864858	34404	864859	093.4 026.4	20-1611SR-MX-310-2239-09340264-0000A-7	0	18.744	33.4083	5.805	23.870
864859	34404	864860	093.4 026.4	20-1611SR-MX-310-2239-09340264-0002A-3	2	18.744	33.4083	5.805	23.870
864860	34404	864861	093.4 026.4	20-1611SR-MX-310-2239-09340264-0005A-3	5	18.692	33.4150	5.796	23.889

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STh
864861	34404	864862	093.4 026.4	20-1611SR-MX-310-2239-09340264-0010A-3	10	18.161	33.4062	5.816	24.014
864862	34404	864863	093.4 026.4	20-1611SR-MX-310-2239-09340264-0015A-3	15	17.533	33.3880	5.774	24.152

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

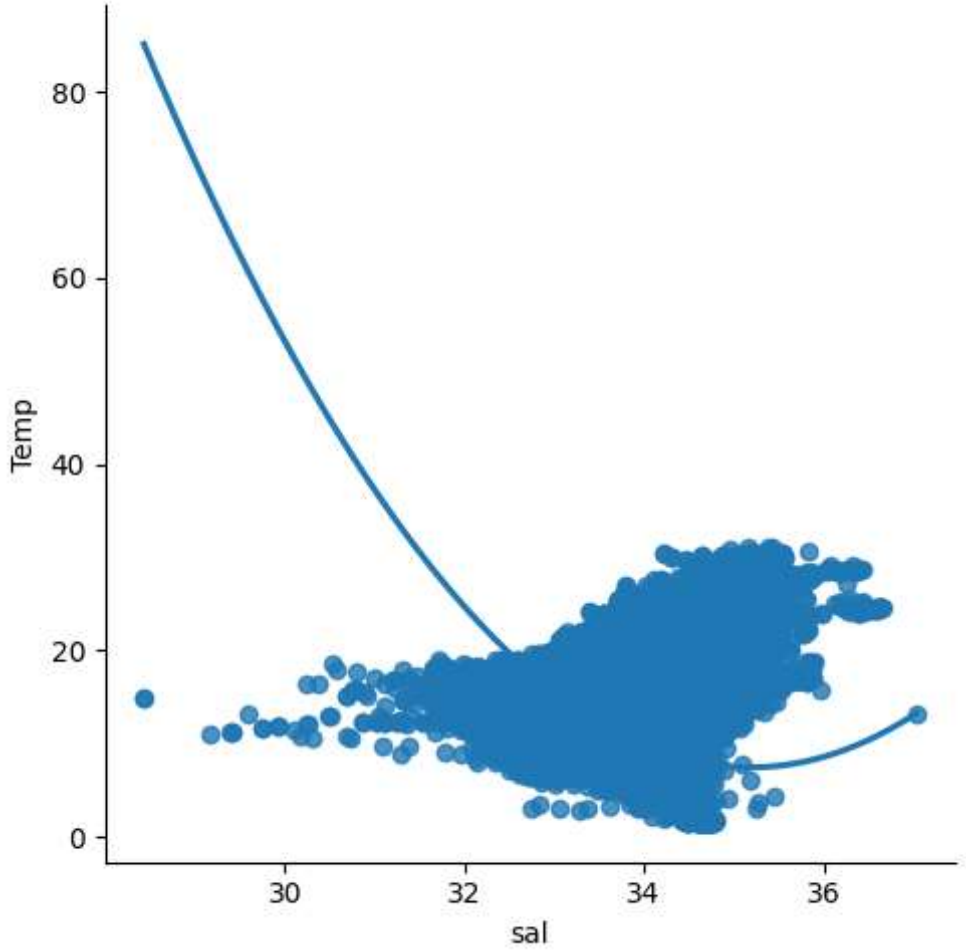
In [5]:
```
df.head(10)
```

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 0x1e51ae1bc10>
```



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 [39]:
```
features=df.columns[0:3]
```

In [40]:
```
target=df.columns[0:2]
```

In [41]:
```
#step 4
df.fillna(method='ffill',inplace=True)
```

```
C:\Users\manis\AppData\Local\Temp\ipykernel_5572\995659311.py:2: SettingWithCopyW
arning:
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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df.fillna(method='ffill',inplace=True)
```

In [42]:
```
#step 5
X=np.array(df['sal']).reshape(-1,1)
y=np.array(df['Temp']).reshape(-1,1)
df.dropna(inplace=True)
```

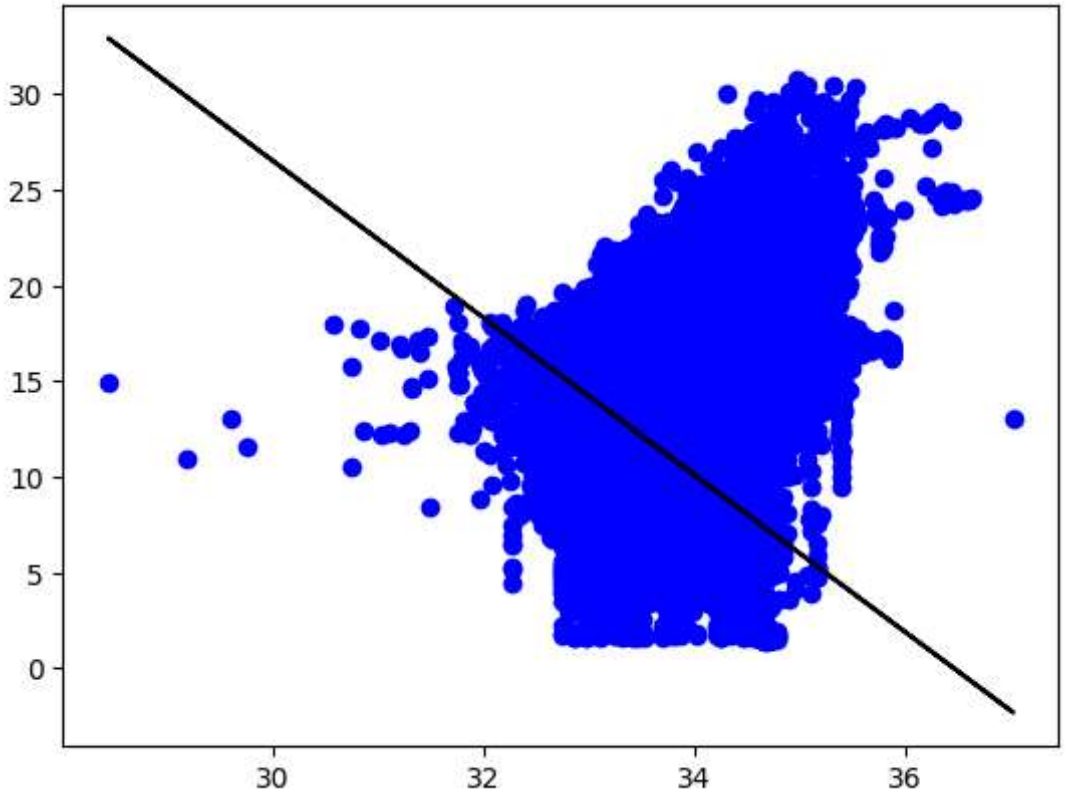
```
C:\Users\manis\AppData\Local\Temp\ipykernel_5572\2811482722.py:4: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df.dropna(inplace=True)
```

In [43]:
```
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))
```

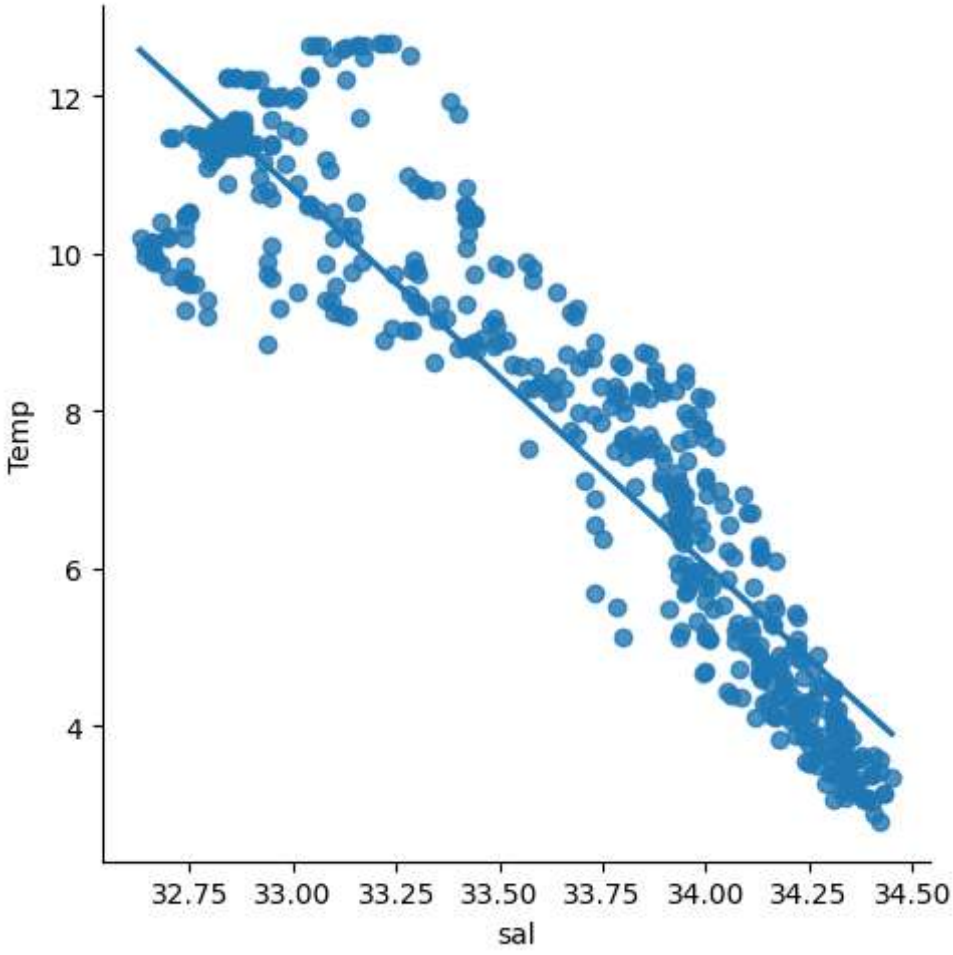
```
0.20564231967230973
```

In [44]:
```
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 [45]:
```
df500=df[:][:500]
sns.lmplot(x='sal',y='Temp',data=df500,order=1,ci=None)
```

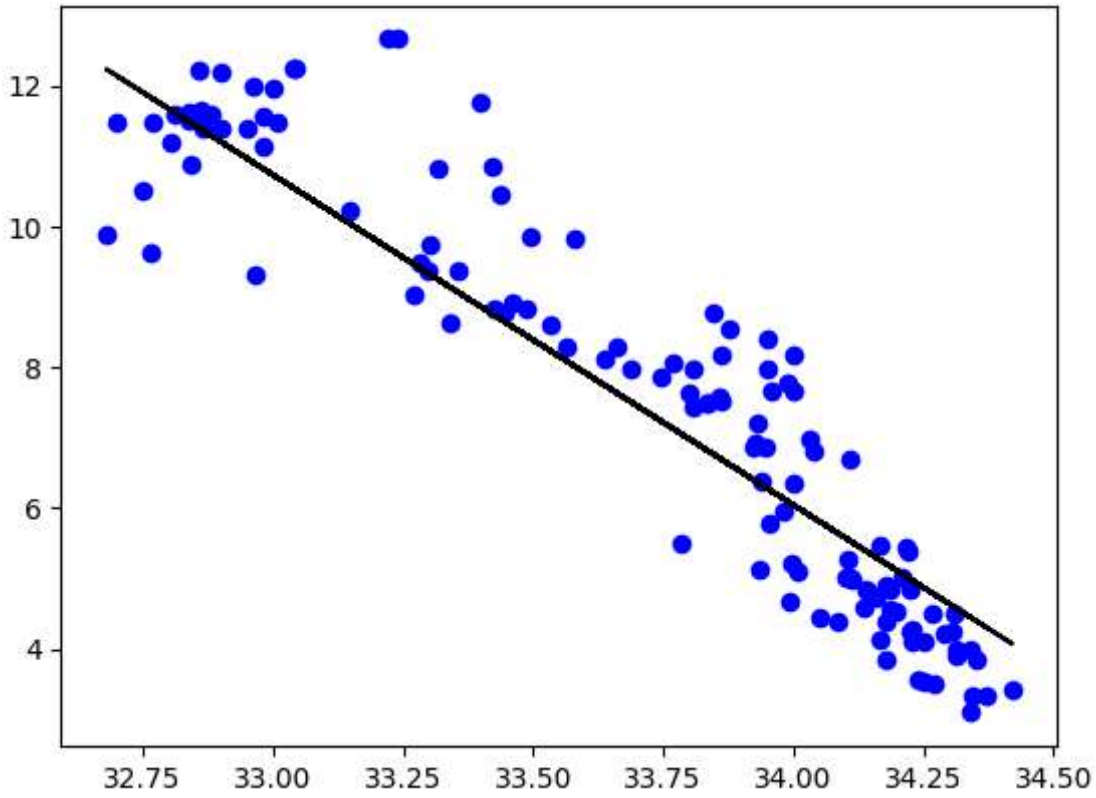
Out[45]:
```
<seaborn.axisgrid.FacetGrid at 0x1e52754fb10>
```



In [46]:
```
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))
```

```
Regression:  0.857910447351786
```

In [47]:
```
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 [48]:
```
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:  0.857910447351786
```

In [49]:
```
from sklearn import metrics
print('MAE:',metrics.mean_absolute_error(y_test,y_pred))
print('MSE:',metrics.mean_squared_error(y_test,y_pred))
print('RMSE:',np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
```

```
MAE: 0.8506902722078349
MSE: 1.1507872845021956
RMSE: 1.0727475399655761
```

ELASTICNET

In [44]:
```
from sklearn.linear_model import ElasticNet
regr=ElasticNet()
regr.fit(X,y)
print(regr.coef_)
print(regr.intercept_)
```

```
[-0.]
[11.41999173]
```

In [45]:
```
y_pred_elastic=regr.predict(X_train)
mean_squared_error=np.mean((y_pred_elastic-y_train)**2)
print("Mean Squared Error on test set",mean_squared_error)
```

```
Mean Squared Error on test set 5.3455007511828905
```

Fiat vehicleselection

(Linear Regression)

In [2]:
```
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 [3]:
```
df=pd.read_csv(r"C:\Users\manis\Downloads\fiat500_VehicleSelection_Dataset.csv")
df
```

Out[3]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	
0	1	lounge	51	882	25000	1	44.907242	
1	2	pop	51	1186	32500	1	45.666359	1
2	3	sport	74	4658	142228	1	45.503300	1
3	4	lounge	51	2739	160000	1	40.633171	1
4	5	pop	73	3074	106880	1	41.903221	1
...	...	...	...	...	...	...	...	
1533	1534	sport	51	3712	115280	1	45.069679	
1534	1535	lounge	74	3835	112000	1	45.845692	
1535	1536	pop	51	2223	60457	1	45.481541	
1536	1537	lounge	51	2557	80750	1	45.000702	
1537	1538	pop	51	1766	54276	1	40.323410	1

1538 rows × 9 columns

In [4]:
```
df=df[['engine_power','lon']]
df.columns=['ep','ln']
```

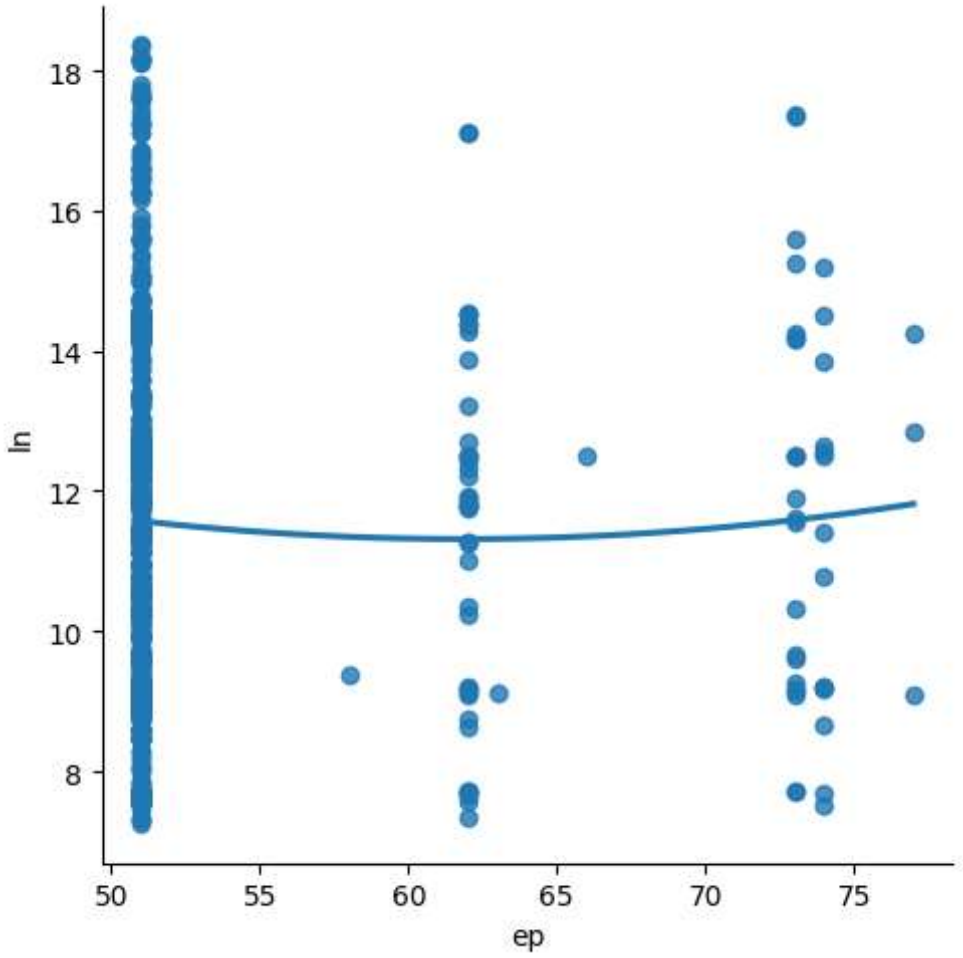
In [5]:
```
df.head(10)
```

Out[5]:

	ep	ln
0	51	8.611560
1	51	12.241890
2	74	11.417840
3	51	17.634609
4	73	12.495650
5	74	7.682270
6	51	8.611560
7	51	12.495650
8	73	11.549470
9	51	10.991700

In [6]:
```
sns.lmplot(x='ep',y='ln',data=df,order=2,ci=None)
```

Out[6]:
```
<seaborn.axisgrid.FacetGrid at 0x1250ecc4c10>
```



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

Out[7]:

	ep	ln
count	1538.000000	1538.000000
mean	51.904421	11.563428
std	3.988023	2.328190
min	51.000000	7.245400
25%	51.000000	9.505090
50%	51.000000	11.869260
75%	51.000000	12.769040
max	77.000000	18.365520

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1538 entries, 0 to 1537
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   ep      1538 non-null   int64  
 1   ln      1538 non-null   float64
dtypes: float64(1), int64(1)
memory usage: 24.2 KB
```

In [29]:
```
df.fillna(method='ffill')
```

Out[29]:

	ep	ln
0	51	8.611560
1	51	12.241890
2	74	11.417840
3	51	17.634609
4	73	12.495650
...	...	...
1533	51	7.704920
1534	74	8.666870
1535	51	9.413480
1536	51	7.682270
1537	51	17.568270

1538 rows × 2 columns

In [30]:
```
X=np.array(df['ep']).reshape(-1,1)
y=np.array(df['ln']).reshape(-1,1)
df.dropna()
```

Out[30]:

	ep	ln
0	51	8.611560
1	51	12.241890
2	74	11.417840
3	51	17.634609
4	73	12.495650
...	...	...
1533	51	7.704920
1534	74	8.666870
1535	51	9.413480
1536	51	7.682270
1537	51	17.568270

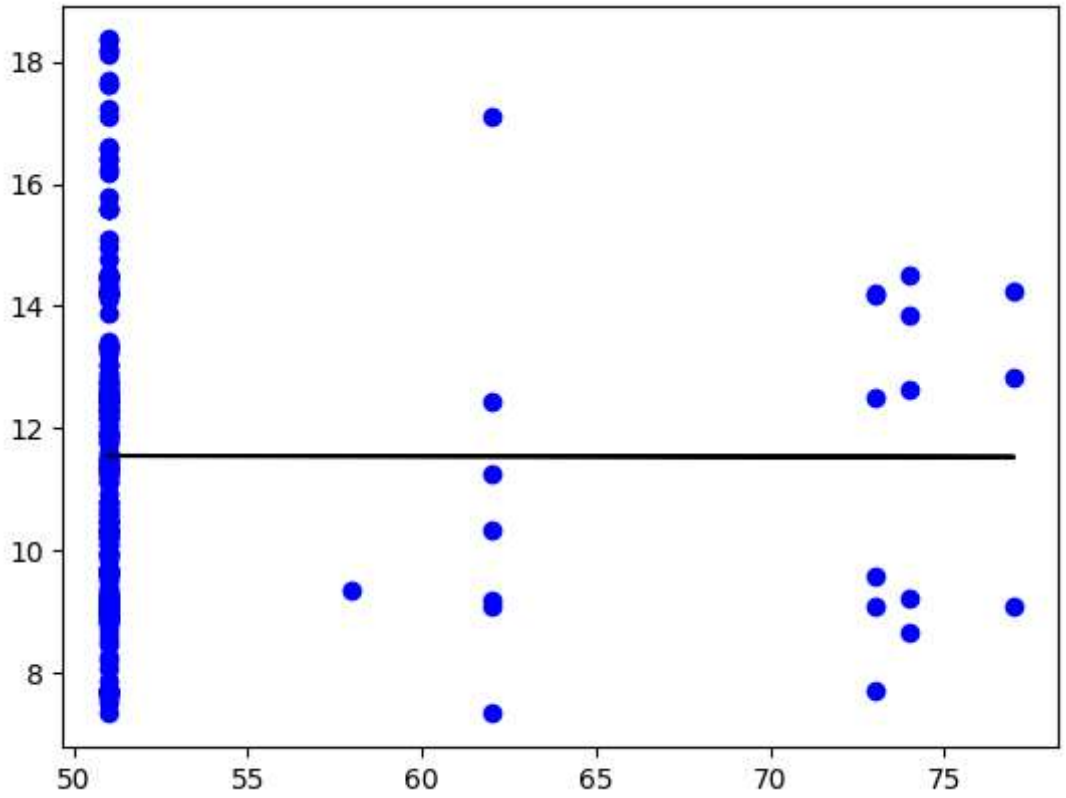
1538 rows × 2 columns

In [31]:
```
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))
```

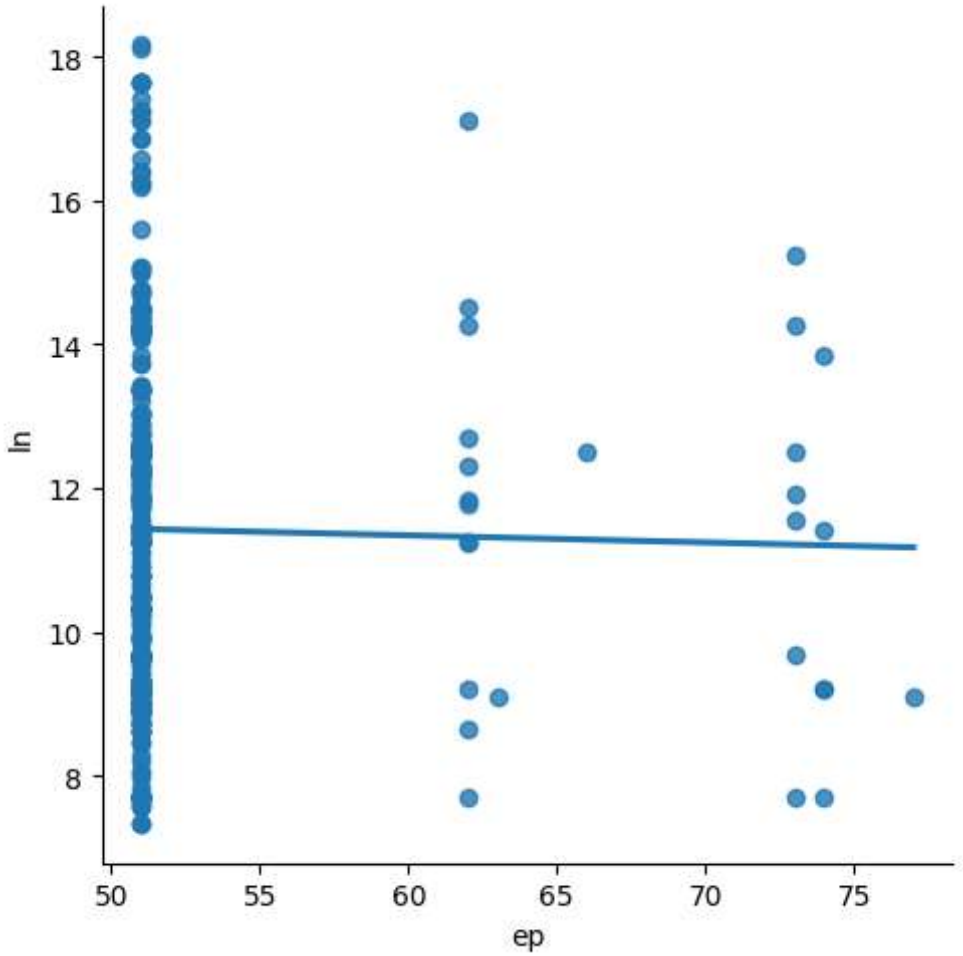
```
-0.0005267394943673231
```

In [32]:
```
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 [33]:
```
df500=df[:][:500]
sns.lmplot(x='ep',y='ln',data=df500,order=1,ci=None)
```

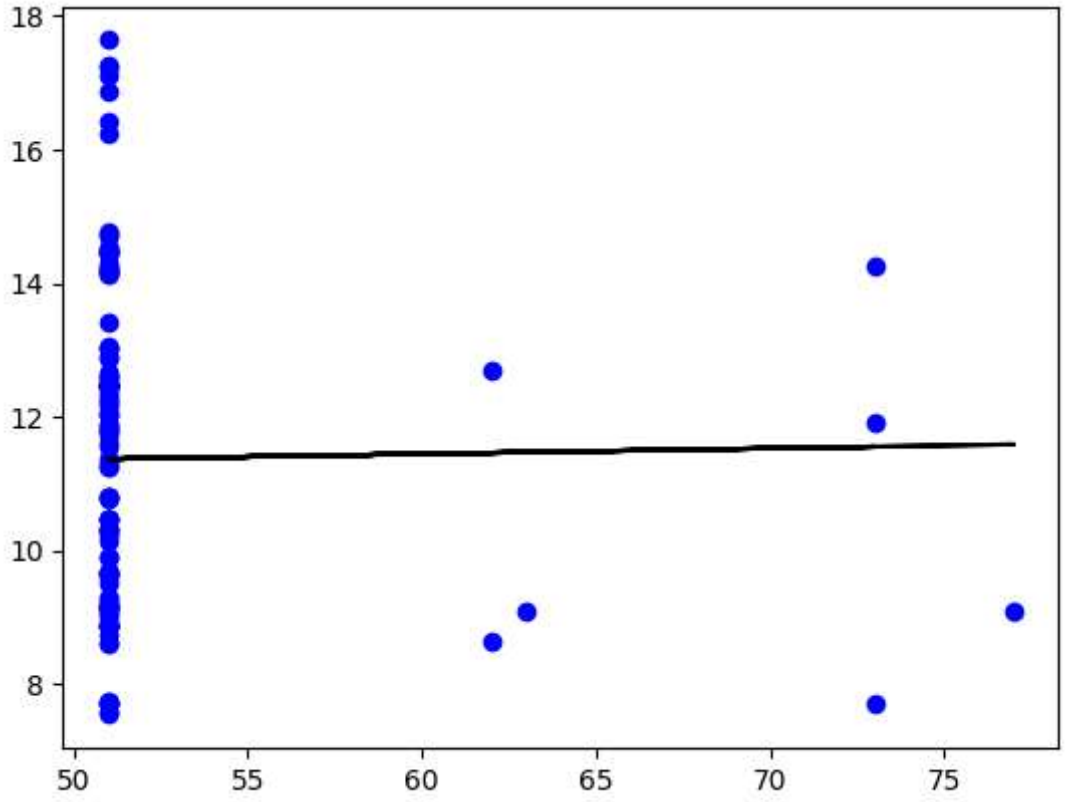
Out[33]:
```
<seaborn.axisgrid.FacetGrid at 0x1251ed2ca50>
```



In [34]:
```
df500.fillna(method='ffill',inplace=True)
X=np.array(df500['ep']).reshape(-1,1)
y=np.array(df500['ln']).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))
```

```
Regression:  -0.008227512735399456
```

In [35]:
```
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 [36]:
```
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:  -0.008227512735399456
```

In [37]:
```
from sklearn import metrics
print('MAE:',metrics.mean_absolute_error(y_test,y_pred))
print('MSE:',metrics.mean_squared_error(y_test,y_pred))
print('RMSE:',np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
```

```
MAE: 2.0050644383304266
MSE: 5.800706245487672
RMSE: 2.4084655375337367
```

ELASTICNET

In [42]:
```
from sklearn.linear_model import ElasticNet
regr=ElasticNet()
regr.fit(X,y)
print(regr.coef_)
print(regr.intercept_)
```

```
[-0.]
[11.41999173]
```

In [43]:
```
y_pred_elastic=regr.predict(X_train)
mean_squared_error=np.mean((y_pred_elastic-y_train)**2)
print("Mean Squared Error on test set",mean_squared_error)
```

```
Mean Squared Error on test set 5.3455007511828905
```