In [4]:

```
#(SALINITY TEMPERATURE)

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 [5]:

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

```
C:\Users\RAMADEVI SURIPAKA\AppData\Local\Temp\ipykernel_18528\1481418663.p
y: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\RAMADEVI SURIPAKA\Downloads\bottle.csv.zip")
```

Out[5]:

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

In [6]:

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

In [7]:

```
df.head(10)
```

Out[7]:

	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

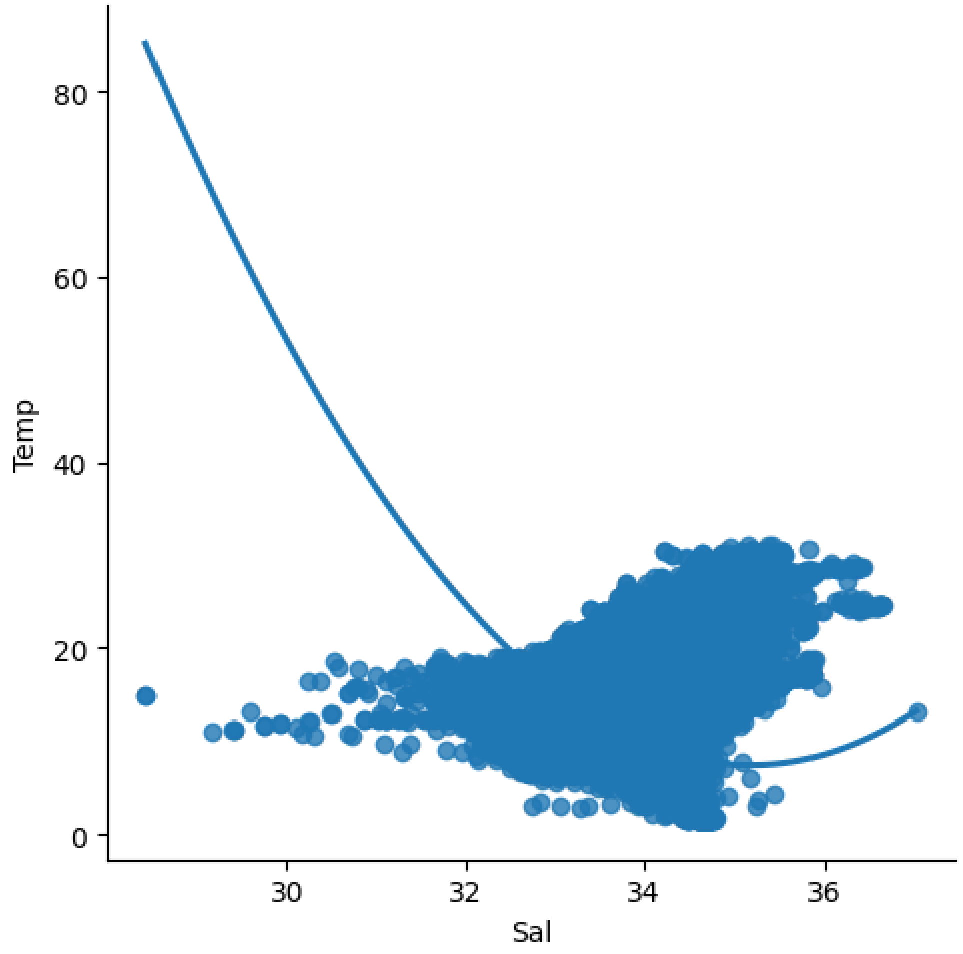
In [12]:

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

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta
864862	34404	864863	093.4 026.4	20-1611SR-MX-310-2239-09340264-	15	17.533	33.3880	5.774	24.15297

Out[12]:

```
<seaborn.axisgrid.FacetGrid at 0x1ec83a4c850>
```



In [13]:

```
df.describe()
```

Out[13]:

	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 [14]:

```
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 [15]:

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

```
C:\Users\RAMADEVI SURIPAKA\AppData\Local\Temp\ipykernel_18528\4116506308.p
y:1: 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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df.fillna(method='ffill',inplace=True)
```

In [18]:

```
x=np.array(df['Sal']).reshape(-1,1)
y=np.array(df['Temp']).reshape(-1,1)
df.dropna(inplace=True)
```

```
C:\Users\RAMADEVI SURIPAKA\AppData\Local\Temp\ipykernel_18528\1516682253.p
y: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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df.dropna(inplace=True)
```

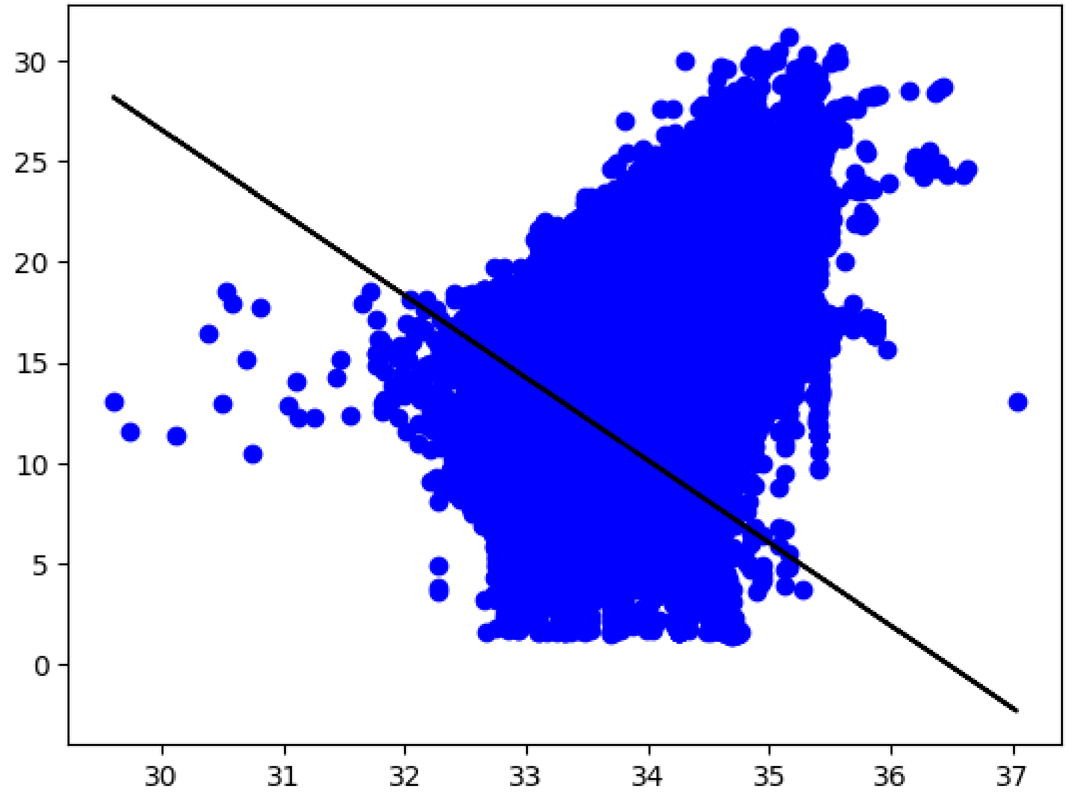
In [20]:

```
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))
```

```
0.20424352864002682
```

In [21]:

```
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```

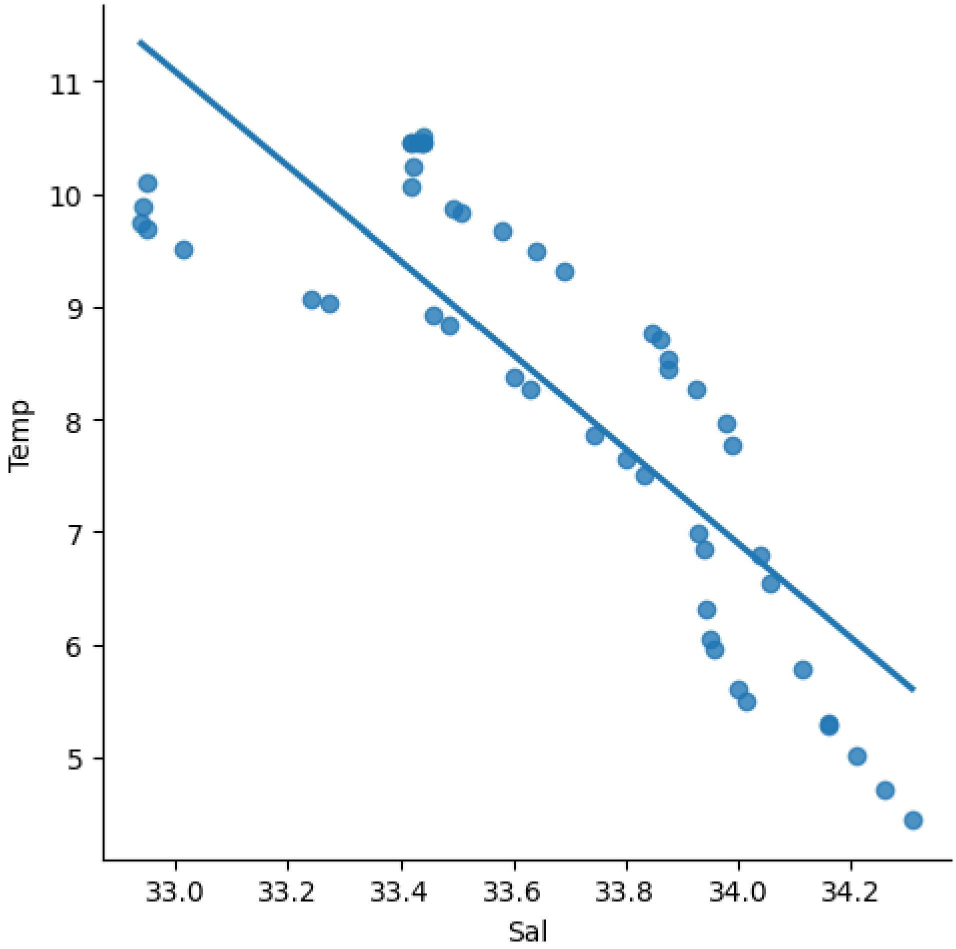


In [24]:

```
df500=df[:][:50]
sns.lmplot(x="Sal",y="Temp",data=df500,order=1,ci=None)
```

Out[24]:

```
<seaborn.axisgrid.FacetGrid at 0x1ec869a6770>
```



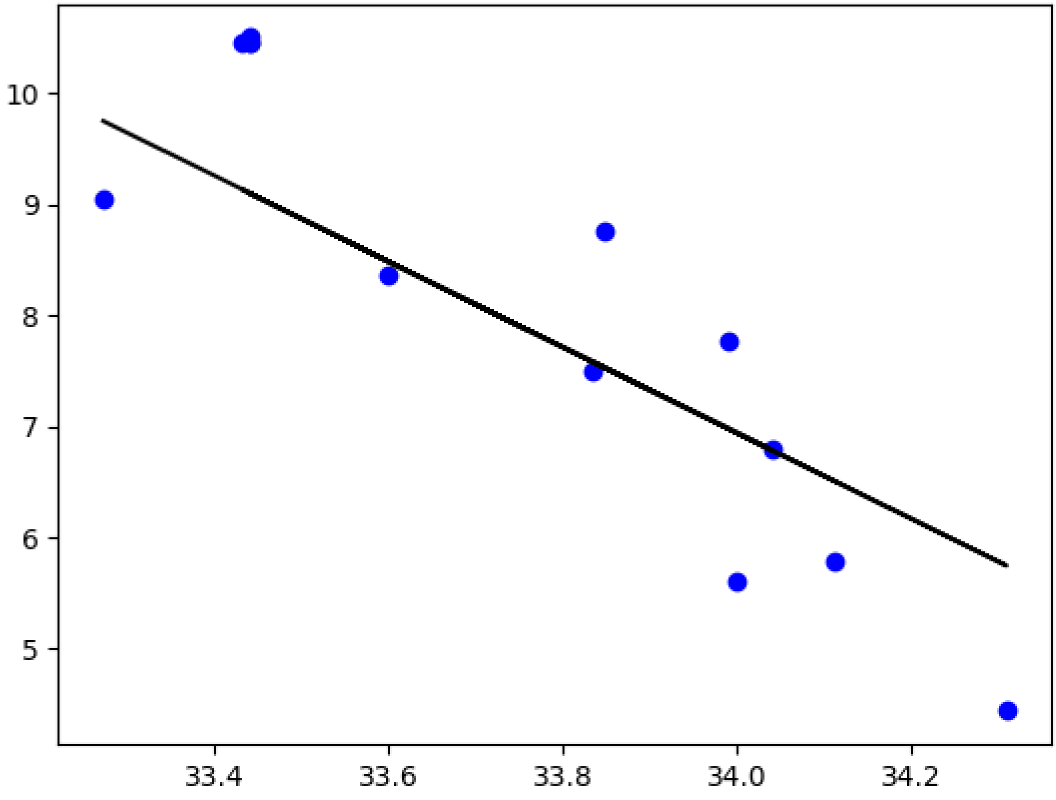
In [26]:

```
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: 0.724238085651293
```

In [27]:

```
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 [28]:

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

In [29]:

```
model=LinearRegression()
model.fit(x_test,y_test)
r2=r2_score(y_test,y_pred)
print("R2 score:",r2)
```

```
R2 score: 0.724238085651293
```

In [1]:

```
#(VEHICLE DATA SET)


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\RAMADEVI SURIPAKA\Downloads\fiat500_VehicleSelection_Dataset (
df
```

Out[2]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	
0	1	lounge	51	882	25000	1	44.907242	8.611
1	2	pop	51	1186	32500	1	45.666359	12.241
2	3	sport	74	4658	142228	1	45.503300	11.417
3	4	lounge	51	2739	160000	1	40.633171	17.634
4	5	pop	73	3074	106880	1	41.903221	12.495
...	...	...	...	...	...	...	...	
1533	1534	sport	51	3712	115280	1	45.069679	7.704
1534	1535	lounge	74	3835	112000	1	45.845692	8.666
1535	1536	pop	51	2223	60457	1	45.481541	9.413
1536	1537	lounge	51	2557	80750	1	45.000702	7.682
1537	1538	pop	51	1766	54276	1	40.323410	17.568

1538 rows × 9 columns

In [16]:

```
df=df[['engine_power','km']]
df.columns=['engine','Km']
```

In [17]:

```
df.head(10)
```

Out[17]:

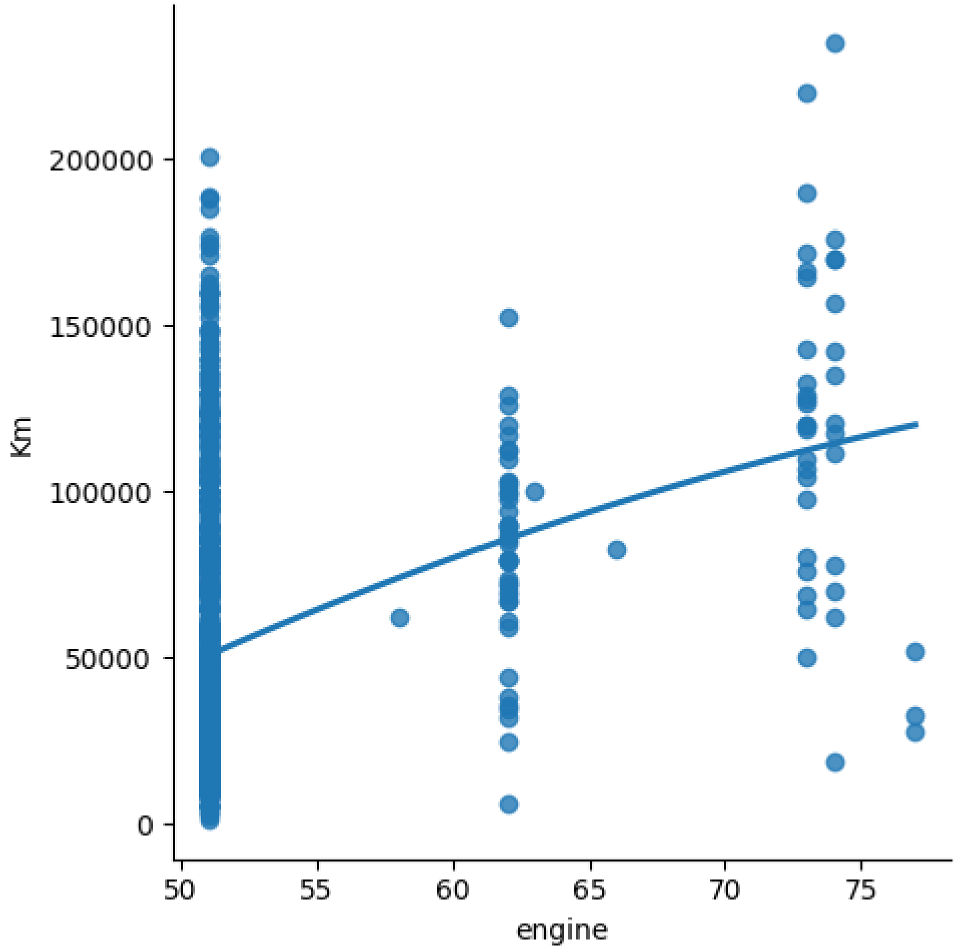
	engine	Km
0	51	25000
1	51	32500
2	74	142228
3	51	160000
4	73	106880
5	74	70225
6	51	11600
7	51	49076
8	73	76000
9	51	89000

In [19]:

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

Out[19]:

```
<seaborn.axisgrid.FacetGrid at 0x1fcb31d6ce0>
```



In [20]:

```
df.describe()
```

Out[20]:

	engine	Km
count	1538.000000	1538.000000
mean	51.904421	53396.011704
std	3.988023	40046.830723
min	51.000000	1232.000000
25%	51.000000	20006.250000
50%	51.000000	39031.000000
75%	51.000000	79667.750000
max	77.000000	235000.000000

In [21]:

```
df.info()
```

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

In [22]:

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

```
C:\Users\RAMADEVI SURIPAKA\AppData\Local\Temp\ipykernel_18236\4116506308.p
y:1: 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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df.fillna(method='ffill',inplace=True)
```

In [23]:

```
x=np.array(df['engine']).reshape(-1,1)
y=np.array(df['Km']).reshape(-1,1)
df.dropna(inplace=True)
```

```
C:\Users\RAMADEVI SURIPAKA\AppData\Local\Temp\ipykernel_18236\821801754.p
y: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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df.dropna(inplace=True)
```

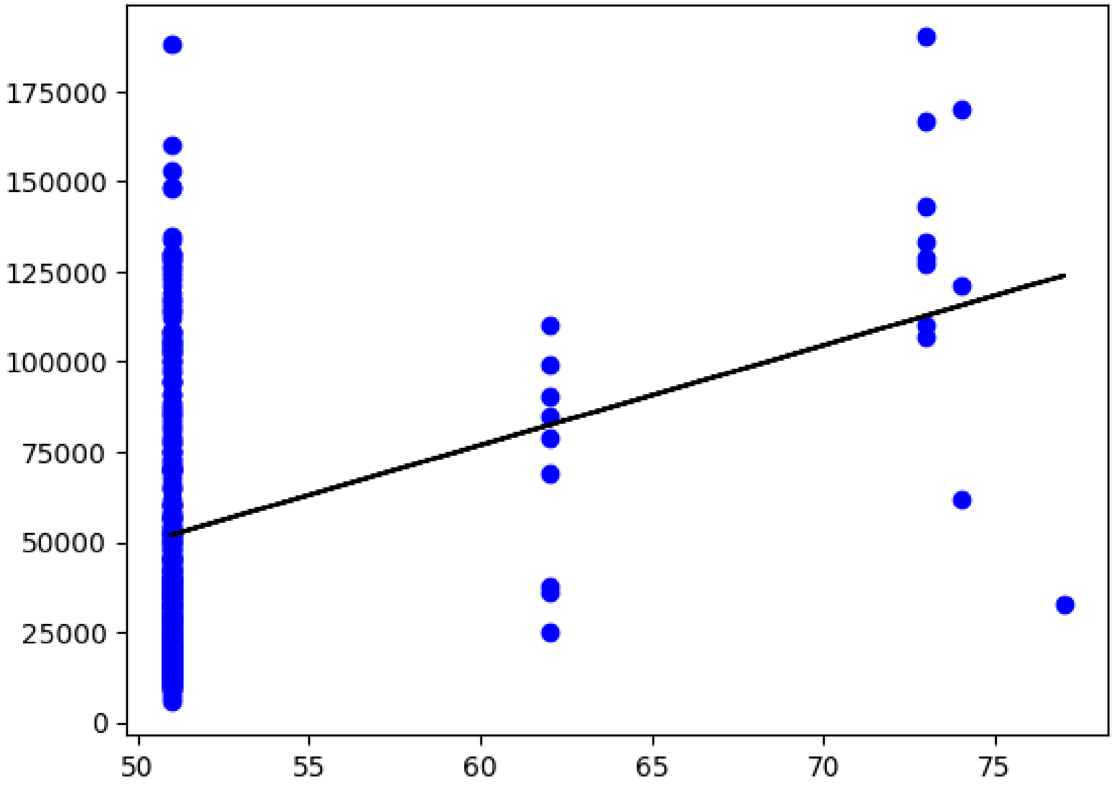
In [24]:

```
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))
```

```
0.10971252862136616
```

In [25]:

```
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```

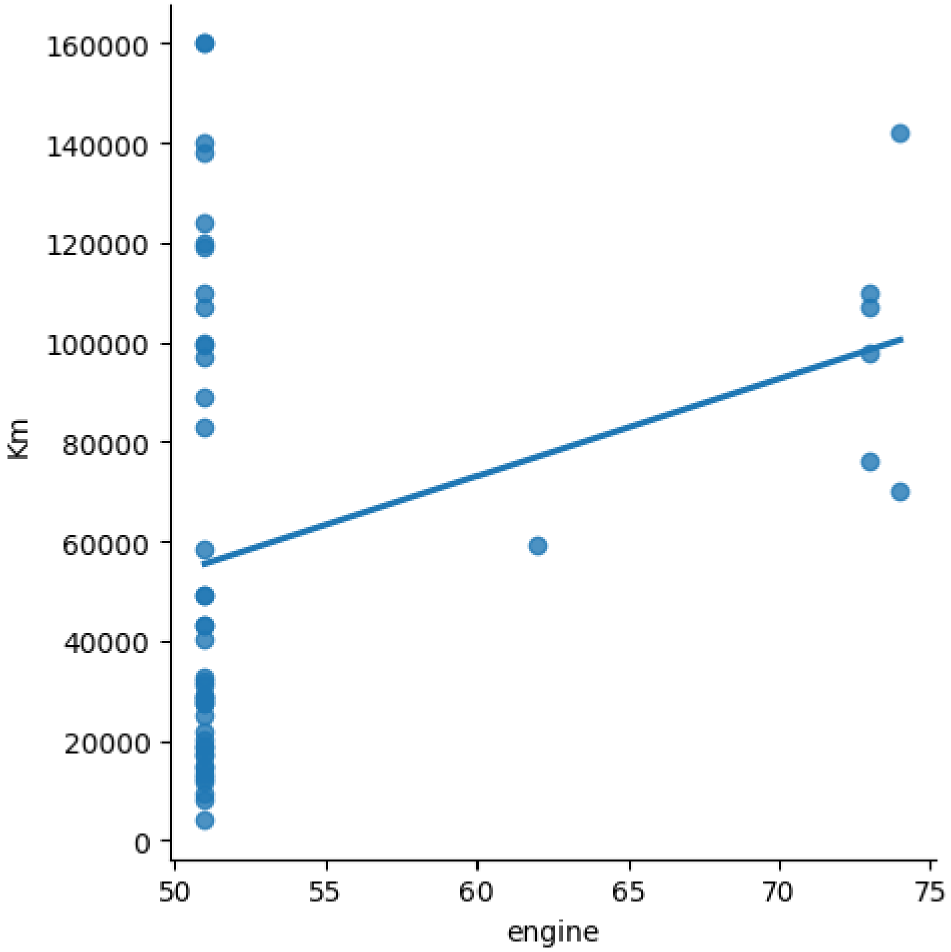


In [26]:

```
df500=df[:][:50]
sns.lmplot(x="engine",y="Km",data=df500,order=1,ci=None)
```

Out[26]:

```
<seaborn.axisgrid.FacetGrid at 0x1fcb34667a0>
```



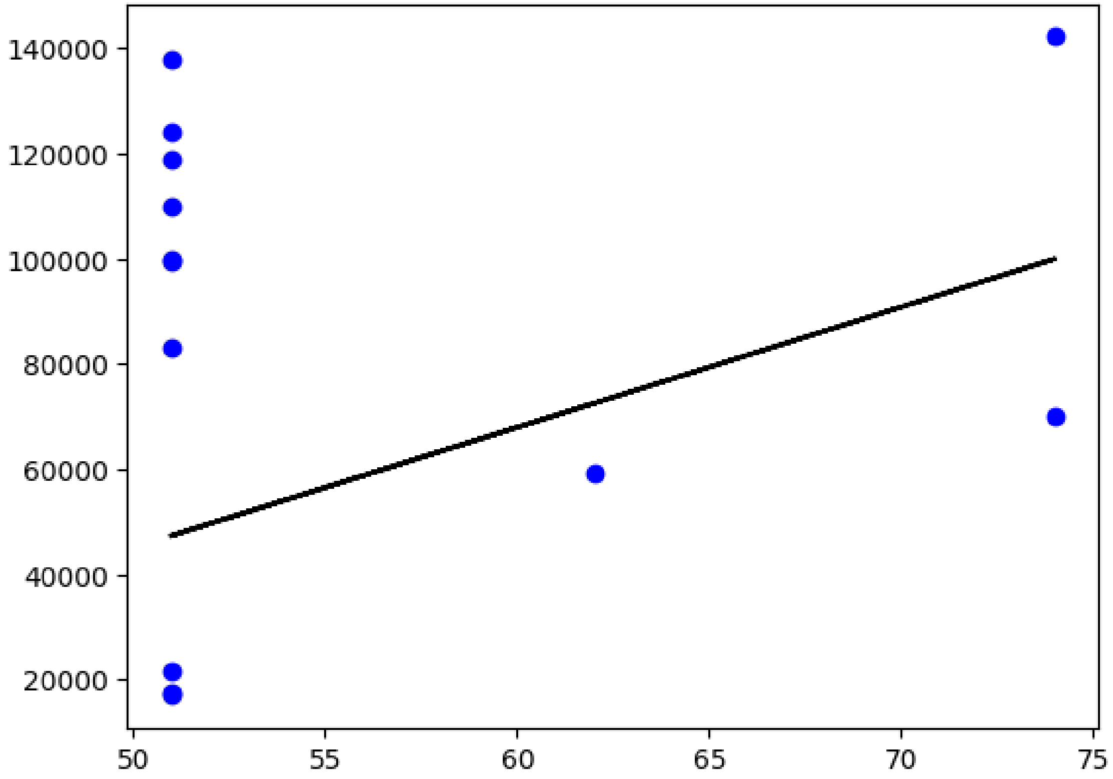
In [27]:

```
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['engine']).reshape(-1,1)
y=np.array(df500['Km']).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: -0.47374063749378 98
```

In [28]:

```
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 [29]:

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

In [30]:

```
model=LinearRegression()
model.fit(x_test,y_test)
r2=r2_score(y_test,y_pred)
print("R2 score:",r2)
```

```
R2 score: -0.4737406374937898
```

In [1]:

```
#(HOUSE PRICE)


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\RAMADEVI SURIPAKA\Downloads\archive.zip")
df
```

Out[2]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwa
0	13300000	7420	4	2	3	yes	no	no	
1	12250000	8960	4	4	4	yes	no	no	
2	12250000	9960	3	2	2	yes	no	yes	
3	12215000	7500	4	2	2	yes	no	yes	
4	11410000	7420	4	1	2	yes	yes	yes	
...	...	...	...	...	...	...	...	...	
540	1820000	3000	2	1	1	yes	no	yes	
541	1767150	2400	3	1	1	no	no	no	
542	1750000	3620	2	1	1	yes	no	no	
543	1750000	2910	3	1	1	no	no	no	
544	1750000	3850	3	1	2	yes	no	no	

545 rows × 13 columns

In [3]:

```
df=df[['price','area']]
df.columns=['Price','Area']
```

In [4]:

```
df.head(10)
```

Out[4]:

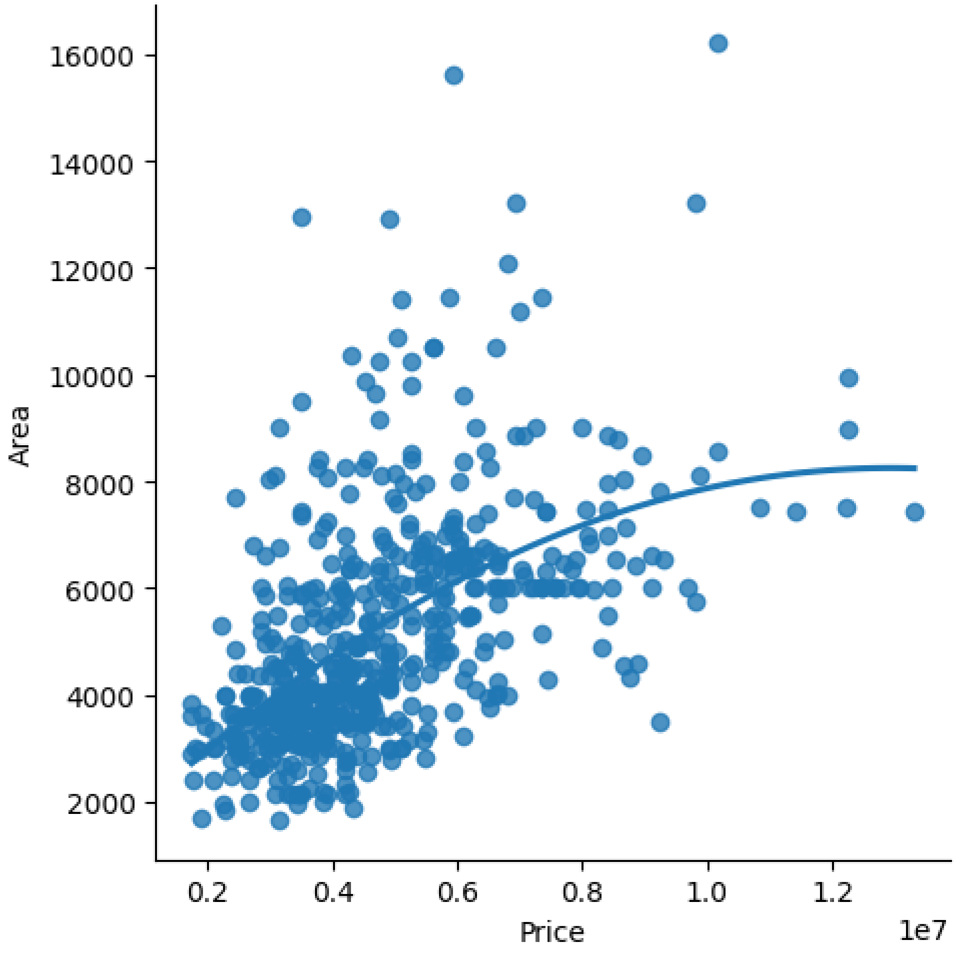
```
      Price   Area
0  13300000   7420
1  12250000   8960
2  12250000   9960
3  12215000   7500
4  11410000   7420
5  10850000   7500
6  10150000   8580
7  10150000  16200
8   9870000   8100
9   9800000   5750
```

In [5]:

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

Out[5]:

```
<seaborn.axisgrid.FacetGrid at 0x280c823e470>
```



In [6]:

```
df.describe()
```

Out[6]:

	Price	Area
count	5.450000e+02	545.000000
mean	4.766729e+06	5150.541284
std	1.870440e+06	2170.141023
min	1.750000e+06	1650.000000
25%	3.430000e+06	3600.000000
50%	4.340000e+06	4600.000000
75%	5.740000e+06	6360.000000
max	1.330000e+07	16200.000000

In [7]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 545 entries, 0 to 544
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Price   545 non-null    int64
 1   Area    545 non-null    int64
dtypes: int64(2)
memory usage: 8.6 KB
```

In [8]:

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

```
C:\Users\RAMADEVI SURIPAKA\AppData\Local\Temp\ipykernel_18500\4116506308.p
y:1: 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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df.fillna(method='ffill',inplace=True)
```

In [9]:

```
x=np.array(df['Price']).reshape(-1,1)
y=np.array(df['Area']).reshape(-1,1)
df.dropna(inplace=True)
```

```
C:\Users\RAMADEVI SURIPAKA\AppData\Local\Temp\ipykernel_18500\3529588183.p
y: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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df.dropna(inplace=True)
```

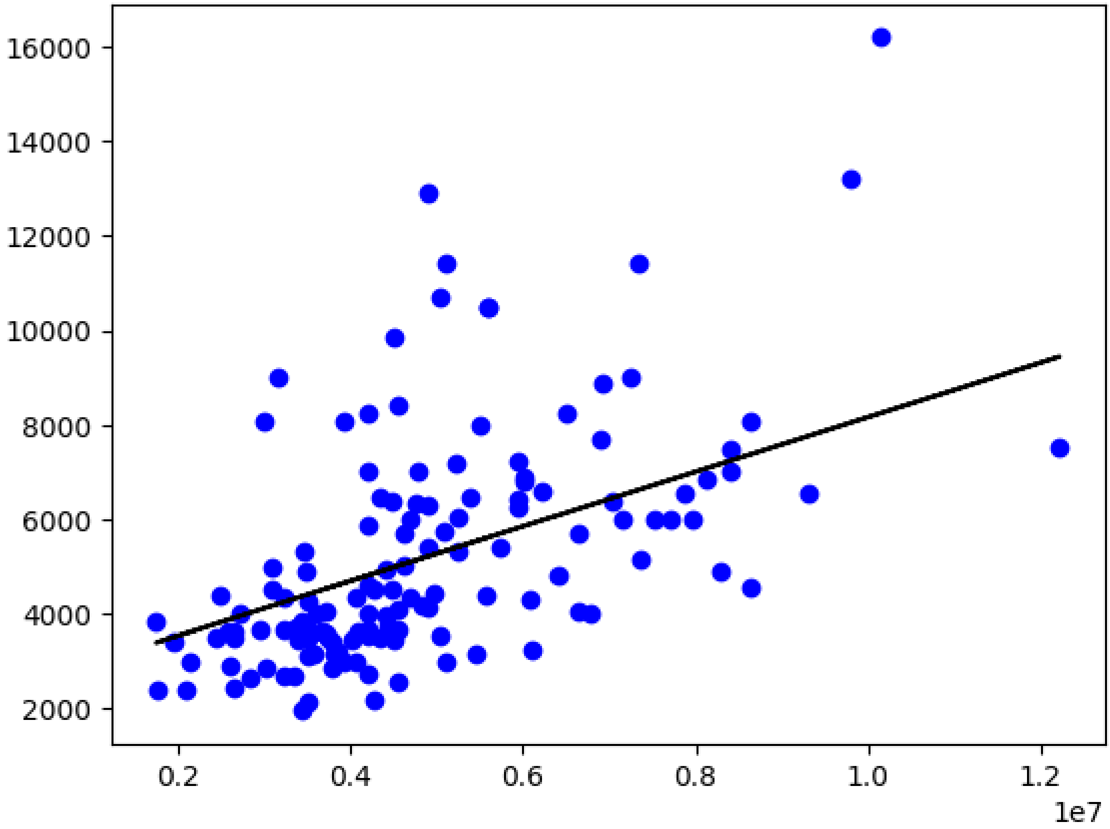
In [10]:

```
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))
```

```
0.2994780602269379
```

In [11]:

```
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```

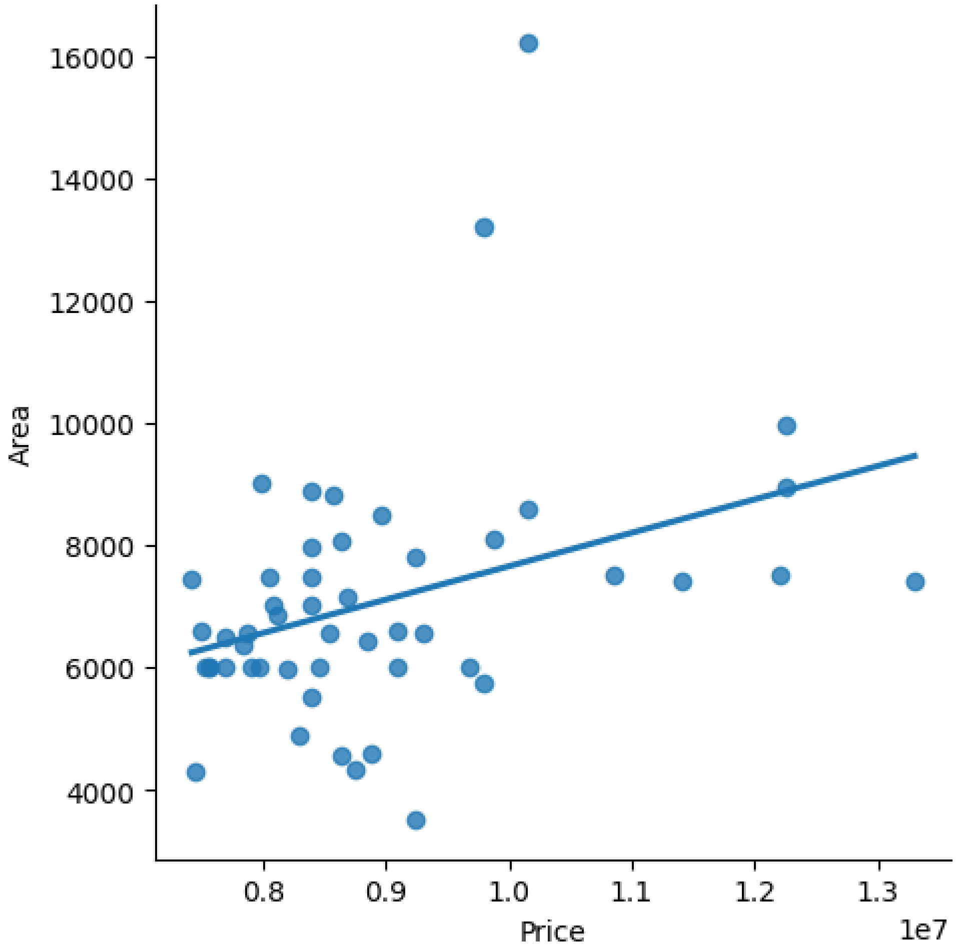


In [12]:

```
df500=df[:][:50]
sns.lmplot(x="Price",y="Area",data=df500,order=1,ci=None)
```

Out[12]:

```
<seaborn.axisgrid.FacetGrid at 0x280b5f31240>
```



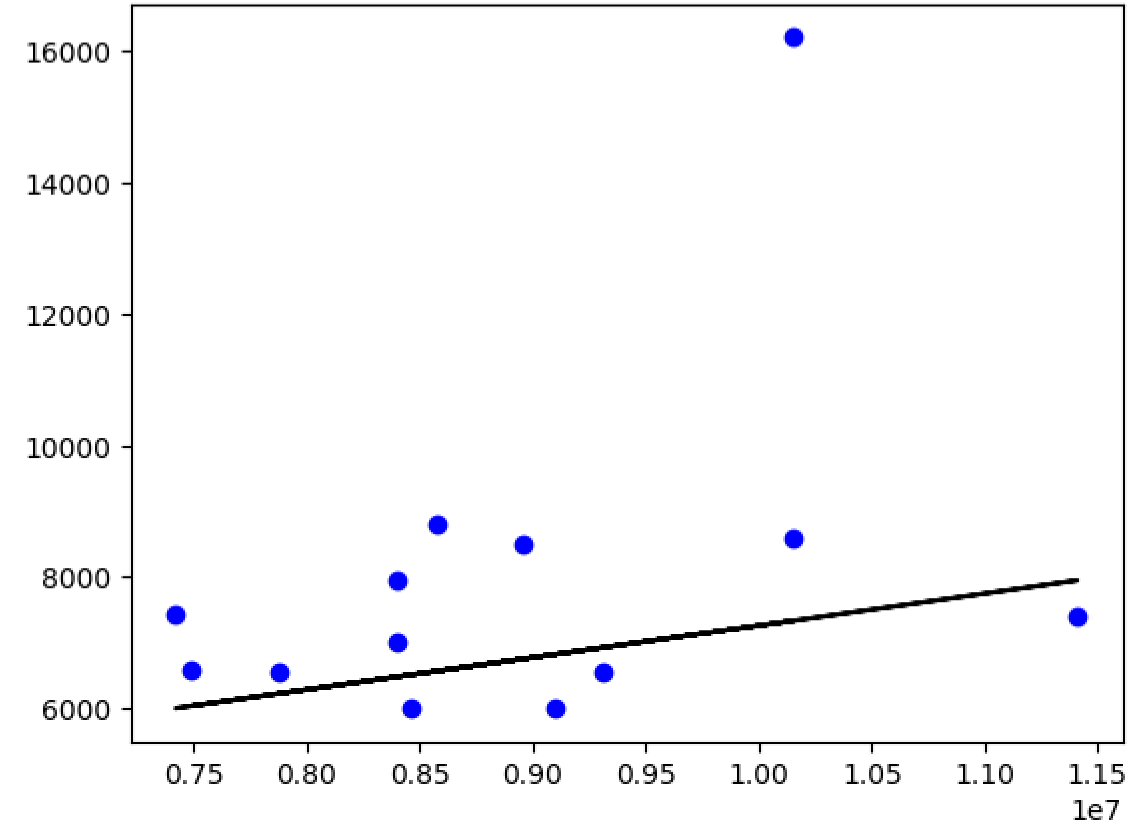
In [13]:

```
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['Price']).reshape(-1,1)
y=np.array(df500['Area']).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: -0.12003962657929845
```

In [14]:

```
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 [15]:

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

In [16]:

```
model=LinearRegression()
model.fit(x_test,y_test)
r2=r2_score(y_test,y_pred)
print("R2 score:",r2)
```

```
R2 score: -0.12003962657929845
```

In [ ]: