In [1]:
```
# importing all the 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
```

bottle dataset

(Linear Regression Model)

In [2]:
```
# reading the file
df=pd.read_csv(r"C:\Users\yoshitha lakshmi\OneDrive\Desktop\python\bottle.csv")
df
```

```
C:\Users\yoshitha lakshmi\AppData\Local\Temp\ipykernel_18836\1239101777.py:2: DtypeWarning: Columns (47,73) have mi
xed types. Specify dtype option on import or set low_memory=False.
  df=pd.read_csv(r"C:\Users\yoshitha lakshmi\OneDrive\Desktop\python\bottle.csv")
```

Out[2]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2Sat	...	R_PHAEO	R_PRES	R_SAMP	DIC1	DIC
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	NaN	NaN	Na
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	NaN	NaN	Na
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	NaN	NaN	Na
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	NaN	NaN	Na
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	NaN	NaN	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	108.74	...	0.18	0	NaN	NaN	Na

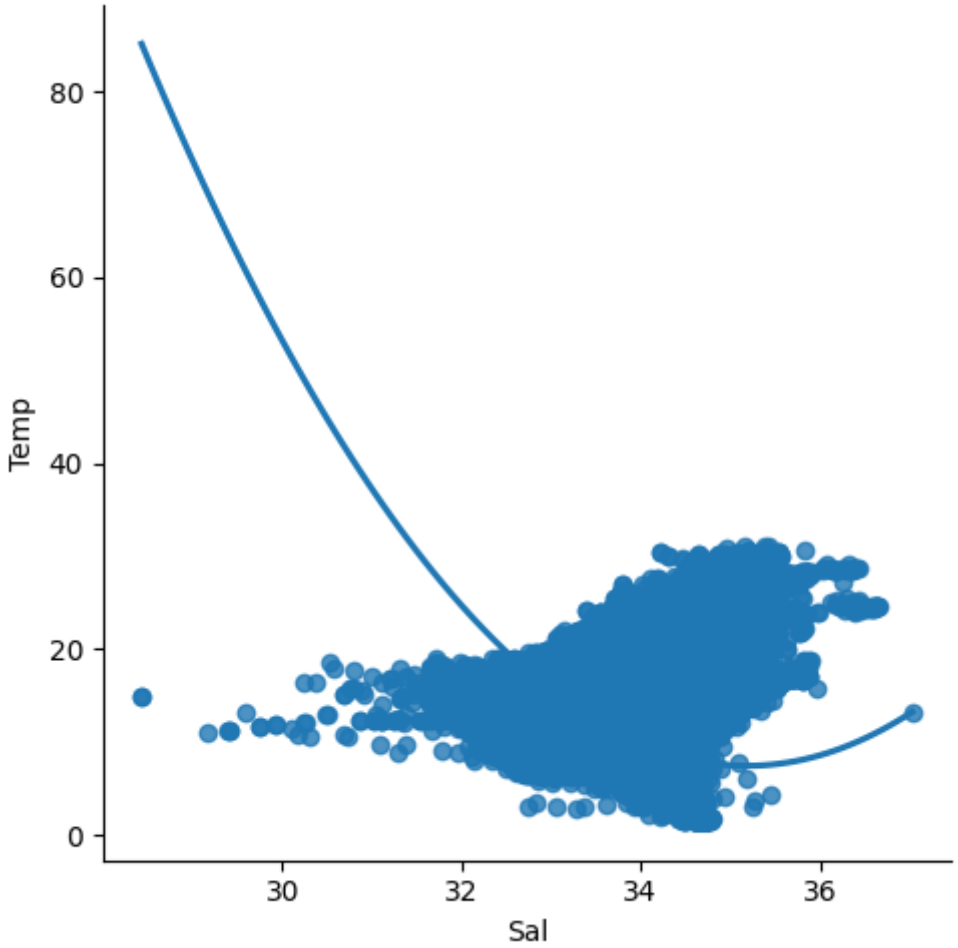
	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2Sat	...	R_PHAEO	R_PRES	R_SAMP	DIC1	DIC
864859	34404	864860	093.4 026.4	20-1611SR-MX-310-2239-09340264-0002A-3	2	18.744	33.4083	5.805	23.87072	108.74	...	0.18	2	4.0	NaN	Na
864860	34404	864861	093.4 026.4	20-1611SR-MX-310-2239-09340264-0005A-3	5	18.692	33.4150	5.796	23.88911	108.46	...	0.18	5	3.0	NaN	Na
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	2.0	NaN	Na
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	1.0	NaN	Na

864863 rows × 74 columns

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

In [4]:
```
# step 3: Exploring the data scatter _plotting the data scatter

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

Out[4]: <seaborn.axisgrid.FacetGrid at 0x25e0a305510>



In [5]:
```
1 df.describe()
```

Out[5]:

	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 [6]:
```
1 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 [7]:
```
# step-4: Data cleaning- Eliminating NaN or missing input numbers

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

```
C:\Users\yoshitha lakshmi\AppData\Local\Temp\ipykernel_18836\524143828.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/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 [8]:
```
# step-5: 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 converting each dataframe
# Now each dataframe contains only one column

```

In [9]:
```
df.dropna(inplace = True)

# Dropping any rows with Nan values
```

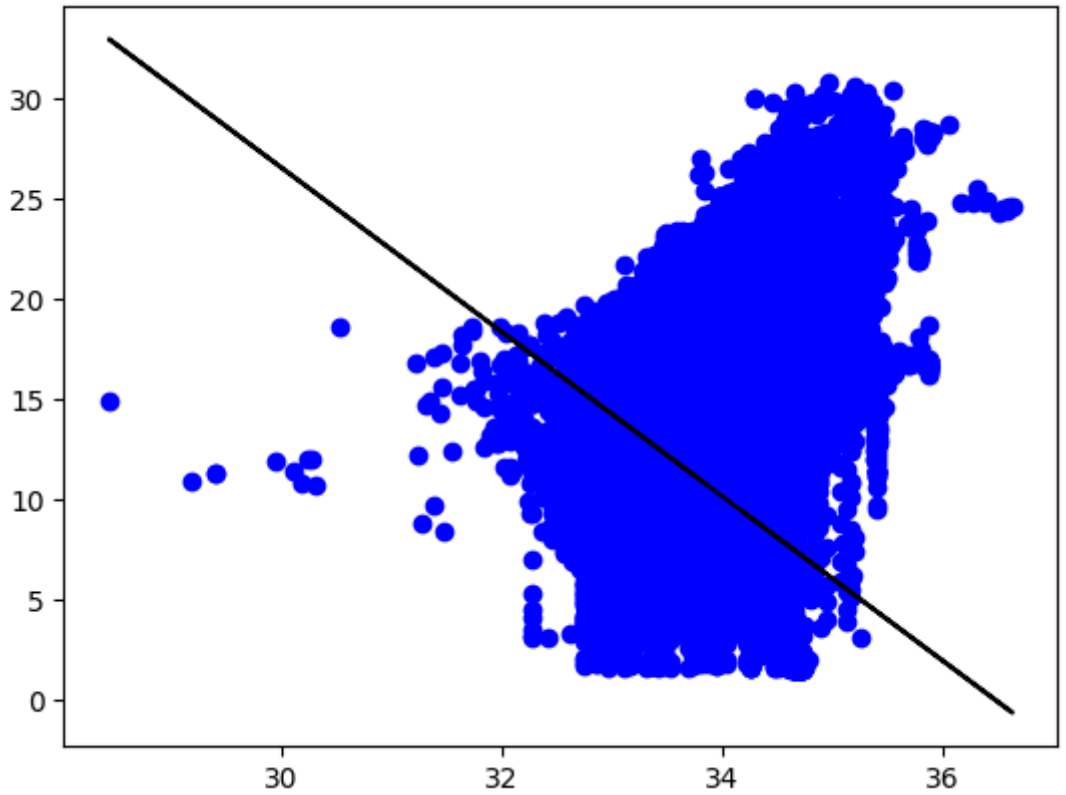
```
C:\Users\yoshitha lakshmi\AppData\Local\Temp\ipykernel_18836\1365071386.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/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)

# Splitting the data into training and testing data

regr = LinearRegression()

regr.fit(X_train, y_train)

print(regr.score(X_test, y_test))
```

```
0.20623479903139552
```

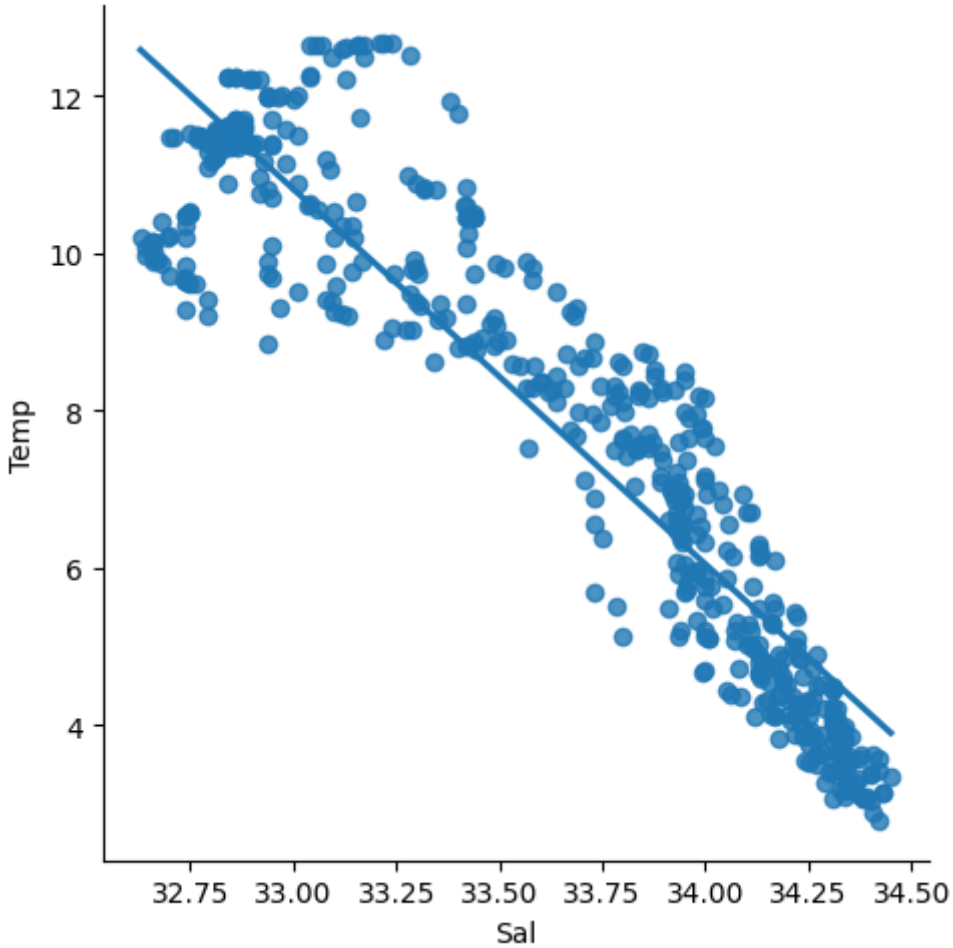
In [11]:

```
# Step-6: 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()

# Data scatter of predicted values
```



In [12]:
```
# step-7: Working with a 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)
```

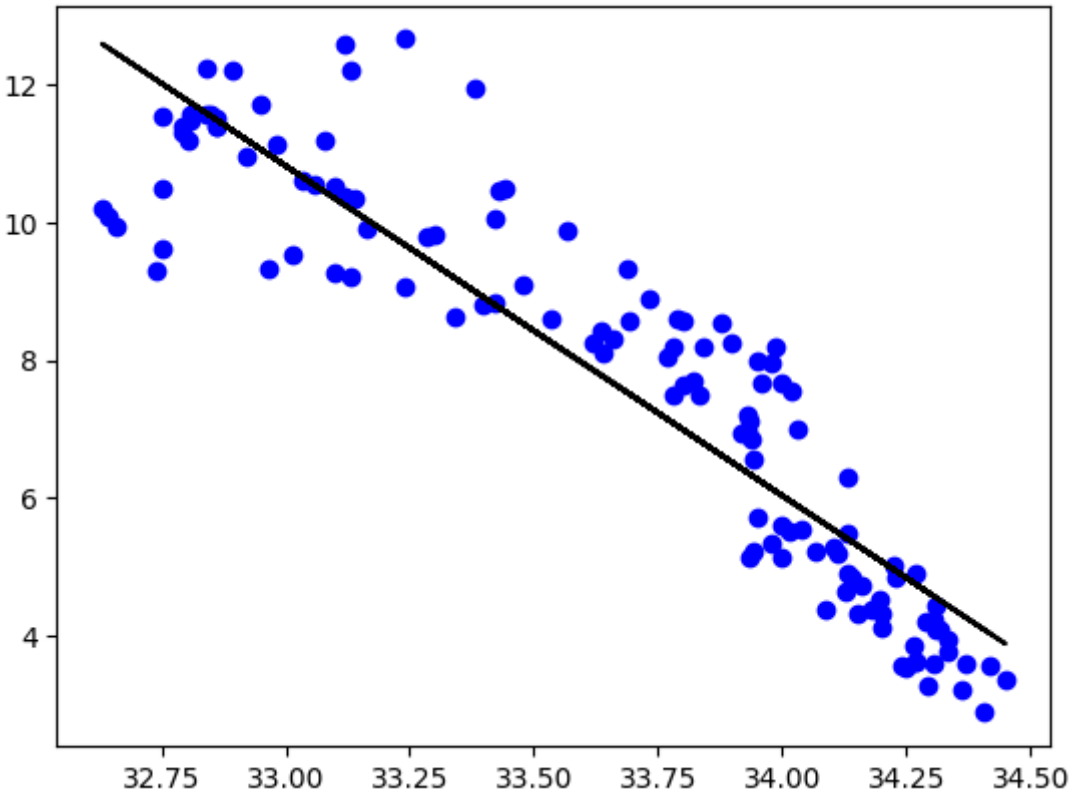
Out[12]: <seaborn.axisgrid.FacetGrid at 0x25e0c290af0>



In [13]:

```
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:  0.8388668884356082
```


12
10
8
6
4
32.75
33.00
33.25
33.50
33.75
34.00
34.25
34.50

In [14]:
```
# Step 8: 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)

y_pred=model.predict(X_test)

r2=r2_score(y_test,y_pred)

print("R2 score: ",r2)
# Evaluate the model on the test set
```

```
R2 score:  0.8388668884356082
```

Conclusion

Dataset we have taken is poor for linear model but with the smaller data works well with linear model.

fiat vehicles dataset

(Linear Regression Model)

In [15]:
```
# importing all the 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 [16]:
```
df=pd.read_csv(r"C:\Users\yoshitha lakshmi\OneDrive\Desktop\python\fiat500_VehicleSelection_Dataset 1.csv")
df
```

Out[16]:

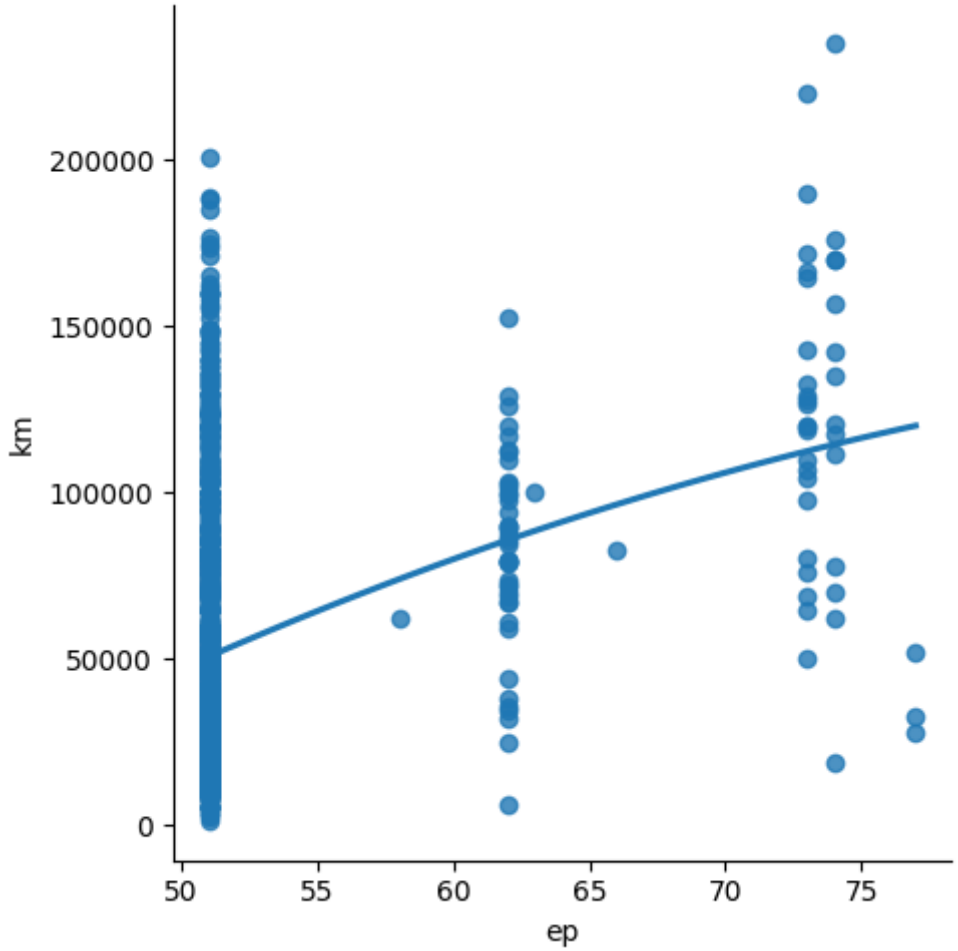
	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	1	lounge	51	882	25000	1	44.907242	8.611560	8900
1	2	pop	51	1186	32500	1	45.666359	12.241890	8800
2	3	sport	74	4658	142228	1	45.503300	11.417840	4200
3	4	lounge	51	2739	160000	1	40.633171	17.634609	6000
4	5	pop	73	3074	106880	1	41.903221	12.495650	5700
...	...	...	...	...	...	...	...	...	...
1533	1534	sport	51	3712	115280	1	45.069679	7.704920	5200
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870	4600
1535	1536	pop	51	2223	60457	1	45.481541	9.413480	7500
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270	5990
1537	1538	pop	51	1766	54276	1	40.323410	17.568270	7900

1538 rows × 9 columns

In [17]:
```
df = df[['engine_power', 'km']]
df.columns=['ep','km']
```

In [18]:
```
sns.lmplot(x="ep", y="km", data=df, order=2, ci= None)
```

Out[18]: <seaborn.axisgrid.FacetGrid at 0x25e0c30c790>



In [19]:
```
1 df.describe()
```

Out[19]:

	ep	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 [20]:
```
1 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   km      1538 non-null   int64
dtypes: int64(2)
memory usage: 24.2 KB
```

In [21]:
```
1 df.fillna(method='ffill', inplace=True)
```

```
C:\Users\yoshitha lakshmi\AppData\Local\Temp\ipykernel_18836\3970806690.py: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-docs/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 [22]:
```
1 X = np.array(df['ep']).reshape(-1,1)
2 y = np.array(df['km']).reshape(-1,1)
3
```

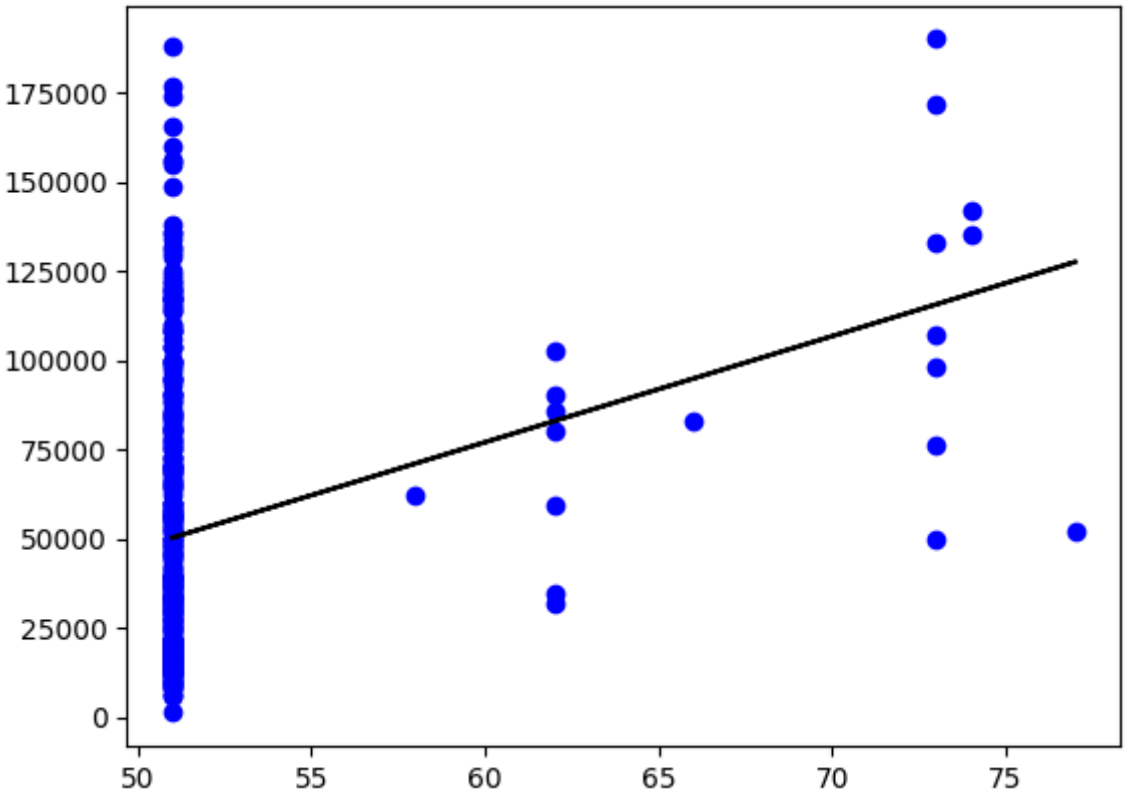
In [23]:
```
1 df.dropna(inplace = True)
```

```
C:\Users\yoshitha lakshmi\AppData\Local\Temp\ipykernel_18836\1791587065.py: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-docs/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]:
```
1 X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.25)
2
3 # Splitting the data into training and testing data
4
5 regr = LinearRegression()
6
7 regr.fit(X_train, y_train)
8
9 print(regr.score(X_test, y_test))
```

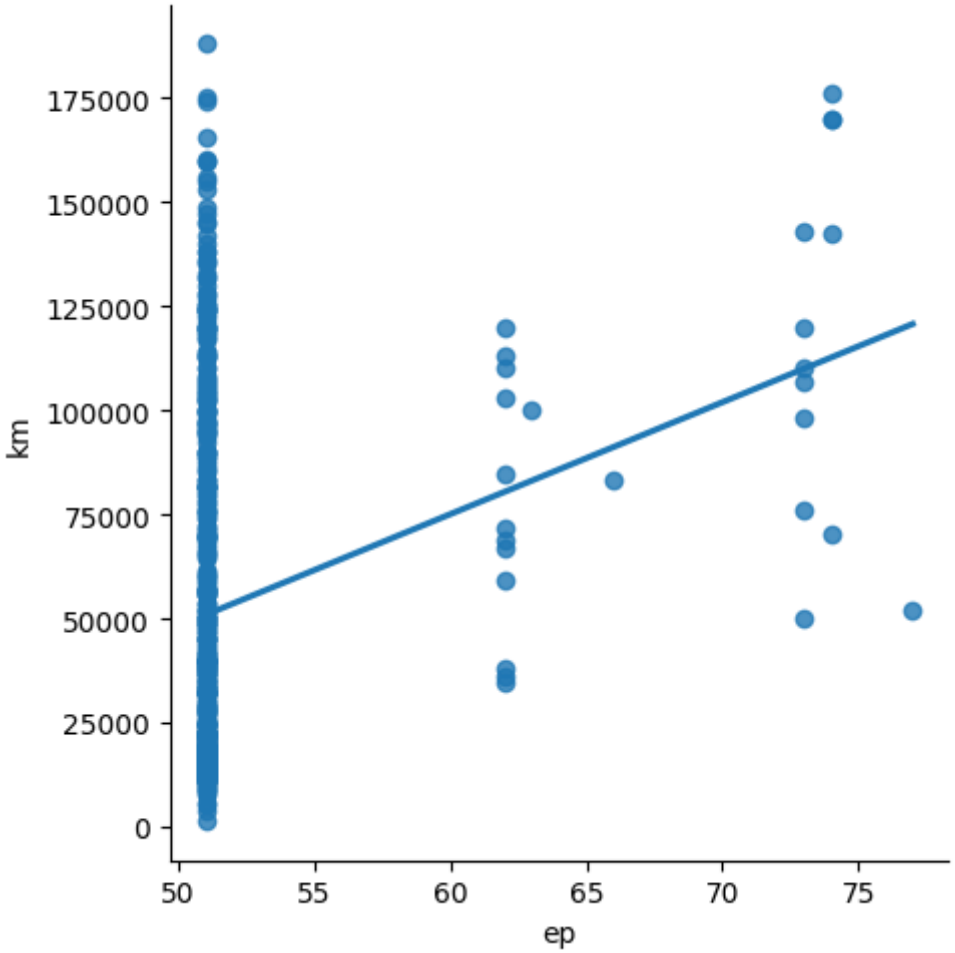
```
0.059743269171622315
```

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[:][:500]

# selecting the 1st 500 rows of the data

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

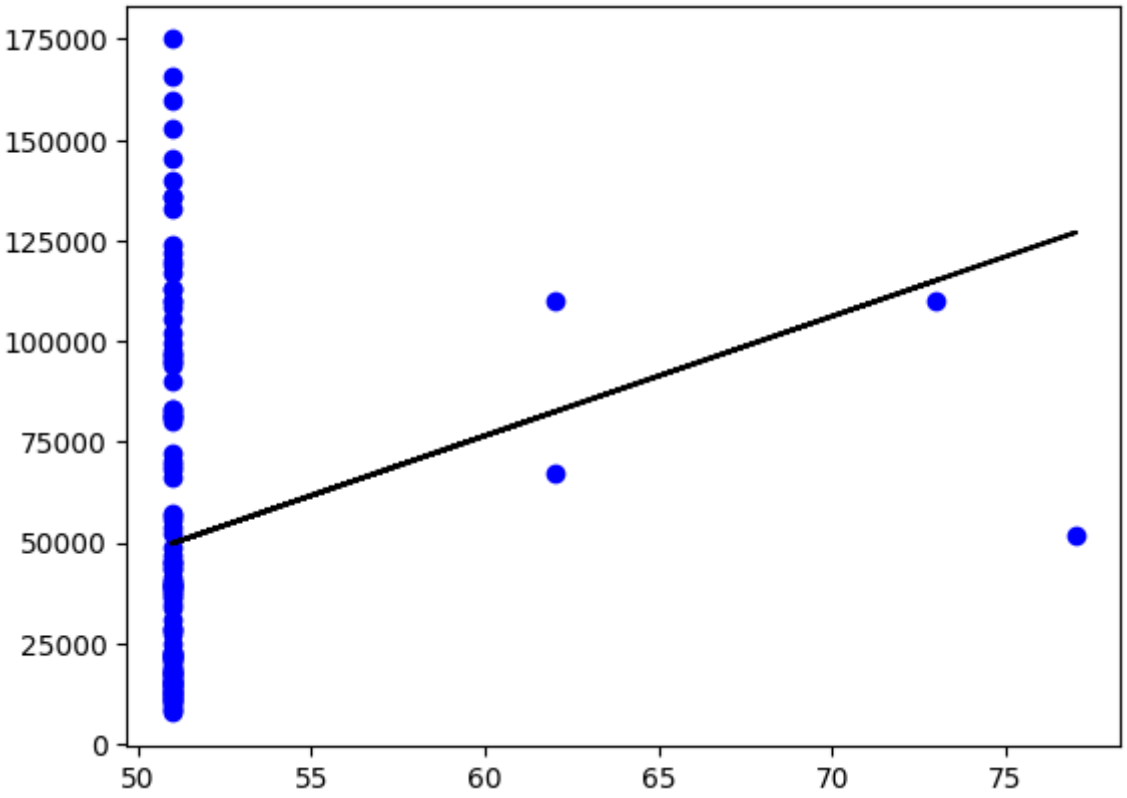
Out[26]: <seaborn.axisgrid.FacetGrid at 0x25e0c30f460>



In [27]:

```
df500.fillna(method ='ffill', inplace = True)

X = np.array(df500['ep']).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))

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:  -0.00849222896402746
```


175000
150000
125000
100000
75000
50000
25000
0
50
55
60
65
70
75

In [28]:
```
# Step 8: 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)

y_pred=model.predict(X_test)

r2=r2_score(y_test,y_pred)

print("R2 score: ",r2)
# Evaluate the model on the test set
```

```
R2 score:  -0.00849222896402746
```

Conclusion

Dataset we have taken is poor for linear model but with the smaller data works well with linear model.

USA House Prediction

In [1]:

```
# importing all the 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\yoshitha lakshmi\OneDrive\Desktop\python\USA_Housing.csv")
df
```

Out[2]:

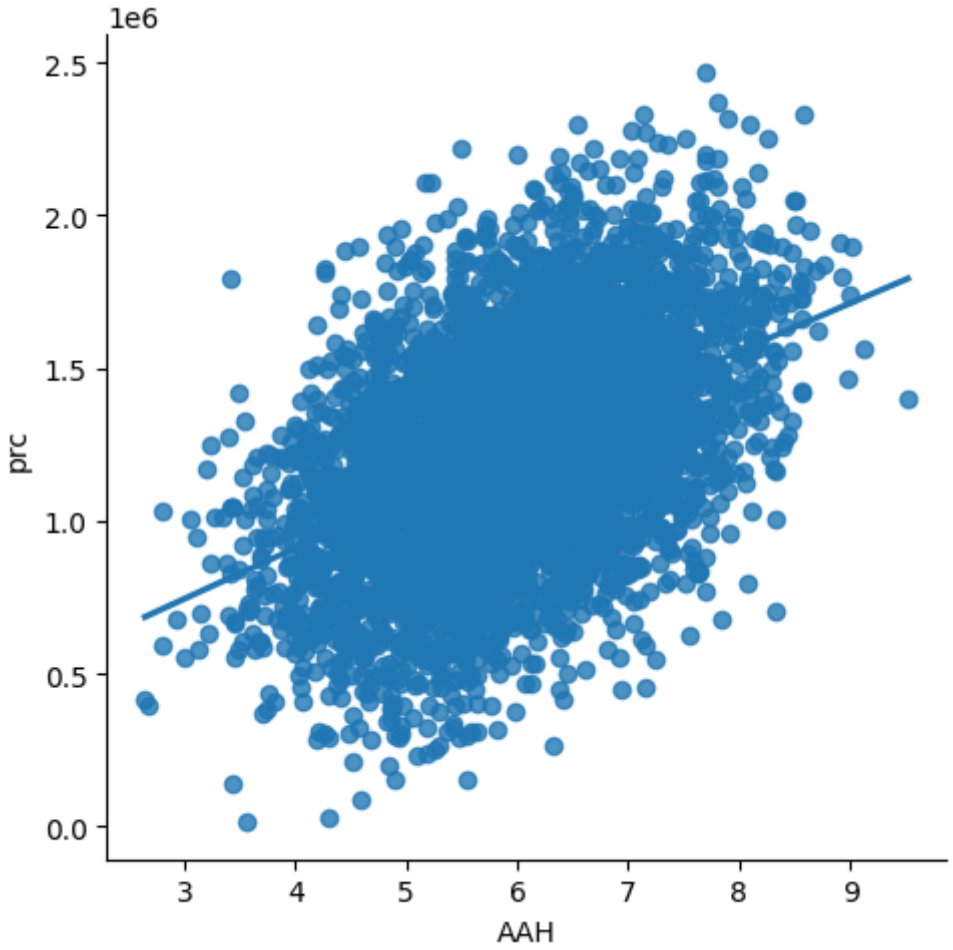
	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	79545.45857	5.682861	7.009188	4.09	23086.80050	1.059034e+06	208 Michael Ferry Apt. 674\nLaurabury, NE 3701...
1	79248.64245	6.002900	6.730821	3.09	40173.07217	1.505891e+06	188 Johnson Views Suite 079\nLake Kathleen, CA...
2	61287.06718	5.865890	8.512727	5.13	36882.15940	1.058988e+06	9127 Elizabeth Stravenue\nDanieltown, WI 06482...
3	63345.24005	7.188236	5.586729	3.26	34310.24283	1.260617e+06	USS Barnett\nFPO AP 44820
4	59982.19723	5.040555	7.839388	4.23	26354.10947	6.309435e+05	USNS Raymond\nFPO AE 09386
...	...	...	...	...	...	...	...
4995	60567.94414	7.830362	6.137356	3.46	22837.36103	1.060194e+06	USNS Williams\nFPO AP 30153-7653
4996	78491.27543	6.999135	6.576763	4.02	25616.11549	1.482618e+06	PSC 9258, Box 8489\nAPO AA 42991-3352
4997	63390.68689	7.250591	4.805081	2.13	33266.14549	1.030730e+06	4215 Tracy Garden Suite 076\nJoshualand, VA 01...
4998	68001.33124	5.534388	7.130144	5.44	42625.62016	1.198657e+06	USS Wallace\nFPO AE 73316
4999	65510.58180	5.992305	6.792336	4.07	46501.28380	1.298950e+06	37778 George Ridges Apt. 509\nEast Holly, NV 2...

5000 rows × 7 columns

In [3]:
```
df = df[['Avg. Area House Age','Price']]
df.columns=['AAH','prc']
```

In [4]:
```
1 sns.lmplot(x='AAH',y='prc',data=df,order=2,ci=None)
```

Out[4]: <seaborn.axisgrid.FacetGrid at 0x1c5a3aa9cc0>



In [5]:
```
1 df.describe()
```

Out[5]:

	AAH	prc
count	5000.000000	5.000000e+03
mean	5.977222	1.232073e+06
std	0.991456	3.531176e+05
min	2.644304	1.593866e+04
25%	5.322283	9.975771e+05
50%	5.970429	1.232669e+06
75%	6.650808	1.471210e+06
max	9.519088	2.469066e+06

In [6]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   AAH     5000 non-null   float64
 1   prc     5000 non-null   float64
dtypes: float64(2)
memory usage: 78.2 KB
```

In [7]:
```
df.fillna(method='ffill',inplace=True)
```

```
C:\Users\yoshitha lakshmi\AppData\Local\Temp\ipykernel_20056\4116506308.py: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-docs/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 [8]:
```
X = np.array(df['AAH']).reshape(-1,1)
y = np.array(df['prc']).reshape(-1,1)
```

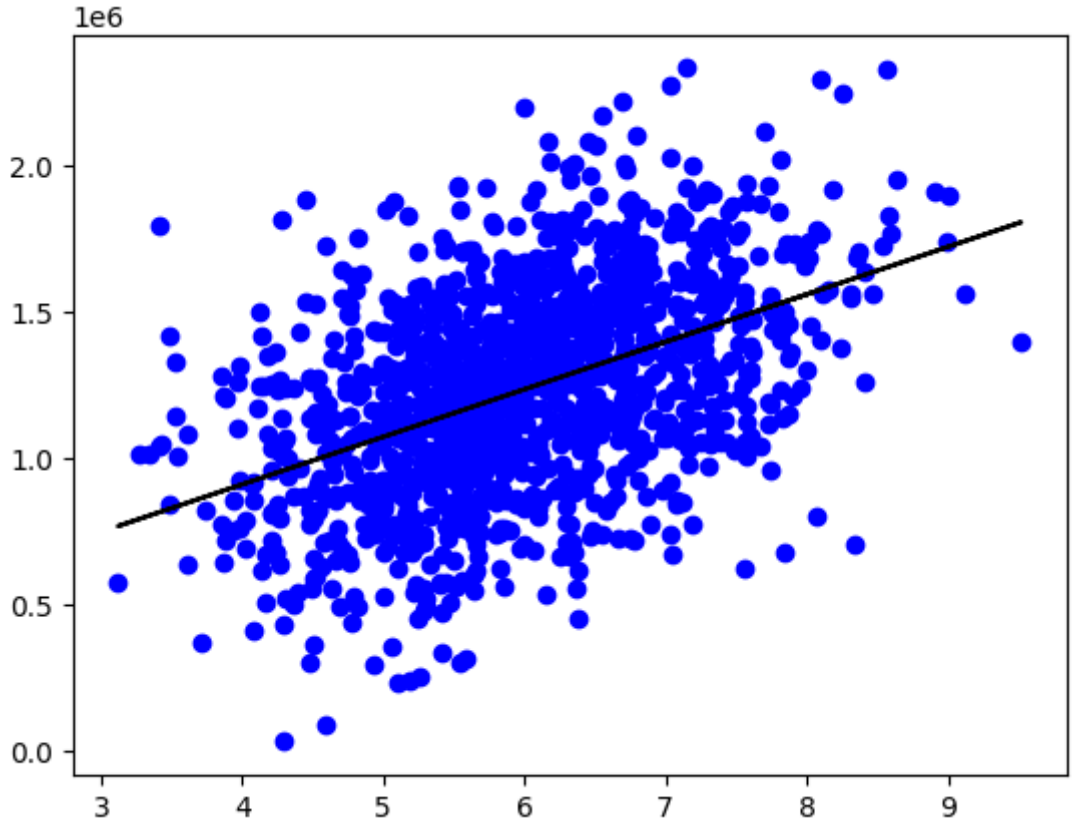
In [9]:
```
df.dropna(inplace = True)
```

```
C:\Users\yoshitha lakshmi\AppData\Local\Temp\ipykernel_20056\1791587065.py: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-docs/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 [11]:
```
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))
```

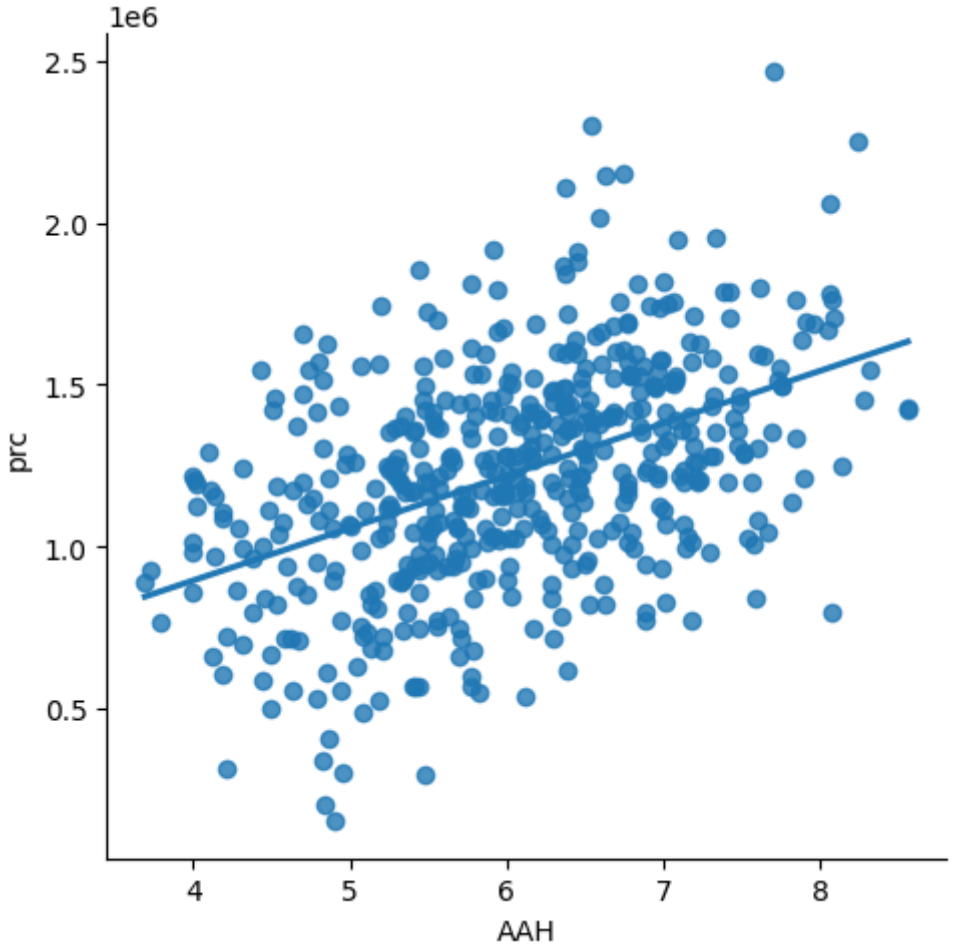
```
0.1986795404108953
```

In [12]:
```
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]:
```
df500 = df[:][:500]
sns.lmplot(x = "AAH", y ="prc", data = df500, order = 1, ci = None)

```

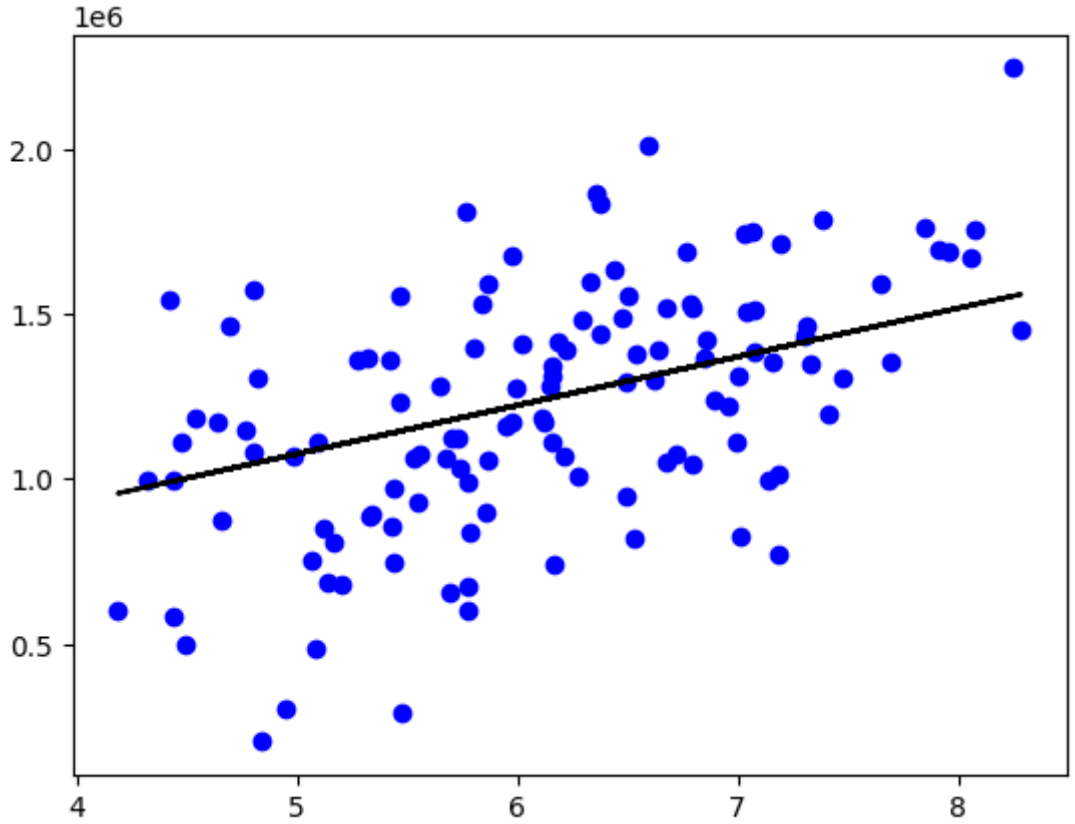
Out[14]: <seaborn.axisgrid.FacetGrid at 0x1c5ab829e40>



In [16]:
```
df500.fillna(method ='ffill', inplace = True)
X = np.array(df500['AAH']).reshape(-1,1)
y = np.array(df500['prc']).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:  0.2656620222707754
```

Out[16]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [17]:

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
# Train the model
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)
# Evaluate the model on the test set
```

```
R2 score:  0.2656620222707754
```

Conclusion

Dataset we have taken is poor for linear model but with the smaller data works well with linear model.

In [ ]:
```
1
```