In [2]:
```
import pandas as pd
import numpy as np
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\HP\Downloads\USA_Housing.csv")
df
```

Out[3]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry Apt. 674\nLaurabury, NE 3701...
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson Views Suite 079\nLake Kathleen, CA...
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizabeth Stravenue\nDanieltown, WI 06482...
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPO AP 44820
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nFPO AE 09386
...	...	...	...	...	...	...	...
4995	60567.944140	7.830362	6.137356	3.46	22837.361035	1.060194e+06	USNS Williams\nFPO AP 30153-7653
4996	78491.275435	6.999135	6.576763	4.02	25616.115489	1.482618e+06	PSC 9258, Box 8489\nAPO AA 42991-3352
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06	4215 Tracy Garden Suite 076\nJoshualand, VA 01...
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06	USS Wallace\nFPO AE 73316
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06	37778 George Ridges Apt. 509\nEast Holly, NV 2...

5000 rows × 7 columns

In [4]:
```
df=df[['Avg. Area Income','Price']]
df.columns=['avg','cost']
```

In [5]:
```
df.info()
```

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

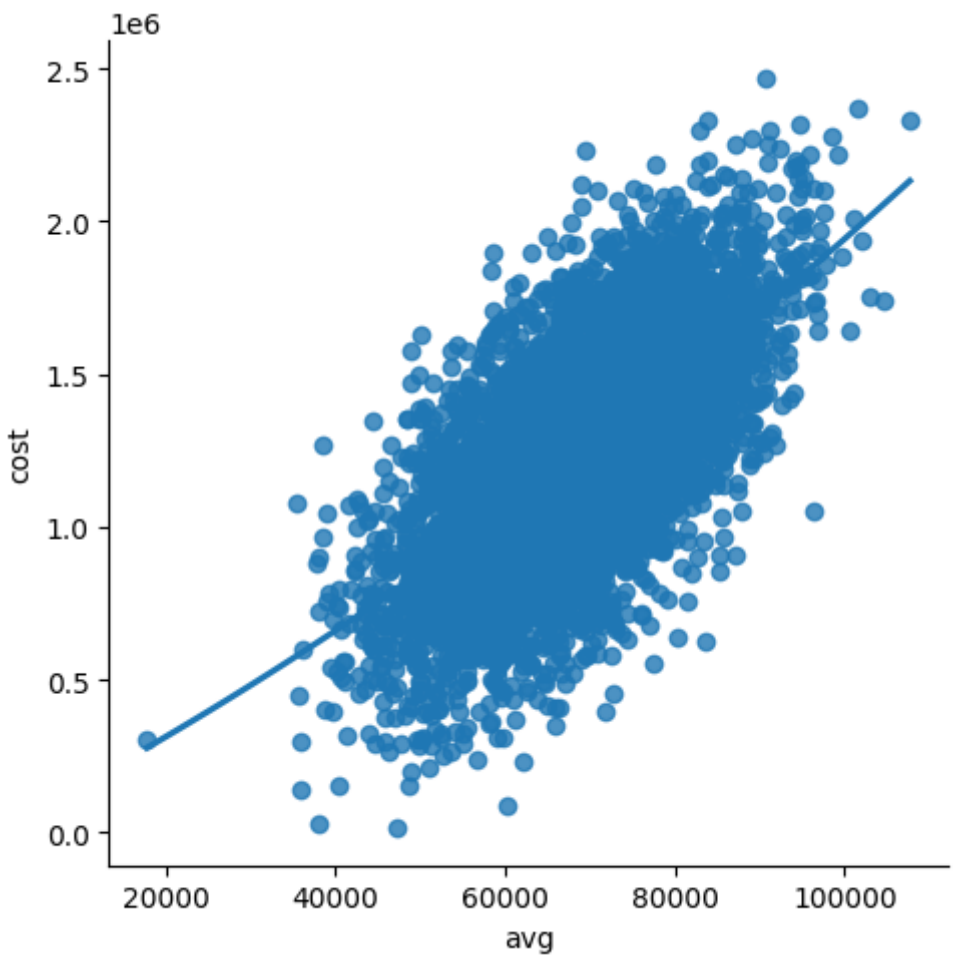
In [6]:
```
df.describe()
```

Out[6]:

	avg	cost
count	5000.000000	5.000000e+03
mean	68583.108984	1.232073e+06
std	10657.991214	3.531176e+05
min	17796.631190	1.593866e+04
25%	61480.562388	9.975771e+05
50%	68804.286404	1.232669e+06
75%	75783.338666	1.471210e+06
max	107701.748378	2.469066e+06

In [7]:
```
1 sns.lmplot(x='avg',y='cost',data=df,order=2,ci=None)
```

Out[7]: <seaborn.axisgrid.FacetGrid at 0x13a5e639ed0>



In [8]:
```
1 df.fillna(method='ffill')
```

Out[8]:

	avg	cost
0	79545.458574	1.059034e+06
1	79248.642455	1.505891e+06
2	61287.067179	1.058988e+06
3	63345.240046	1.260617e+06
4	59982.197226	6.309435e+05
...	...	...
4995	60567.944140	1.060194e+06
4996	78491.275435	1.482618e+06
4997	63390.686886	1.030730e+06
4998	68001.331235	1.198657e+06
4999	65510.581804	1.298950e+06

5000 rows × 2 columns

In [9]:
```
1 df.head(10)
```

Out[9]:

	avg	cost
0	79545.458574	1.059034e+06
1	79248.642455	1.505891e+06
2	61287.067179	1.058988e+06
3	63345.240046	1.260617e+06
4	59982.197226	6.309435e+05
5	80175.754159	1.068138e+06
6	64698.463428	1.502056e+06
7	78394.339278	1.573937e+06
8	59927.660813	7.988695e+05
9	81885.927184	1.545155e+06

In [10]:
```
1 x=np.array(df['avg']).reshape(-1,1)
2 y=np.array(df['cost']).reshape(-1,1)
```

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("Regression: ",regr.score(x_test,y_test))
```

```
Regression:  0.42804390343119125
```

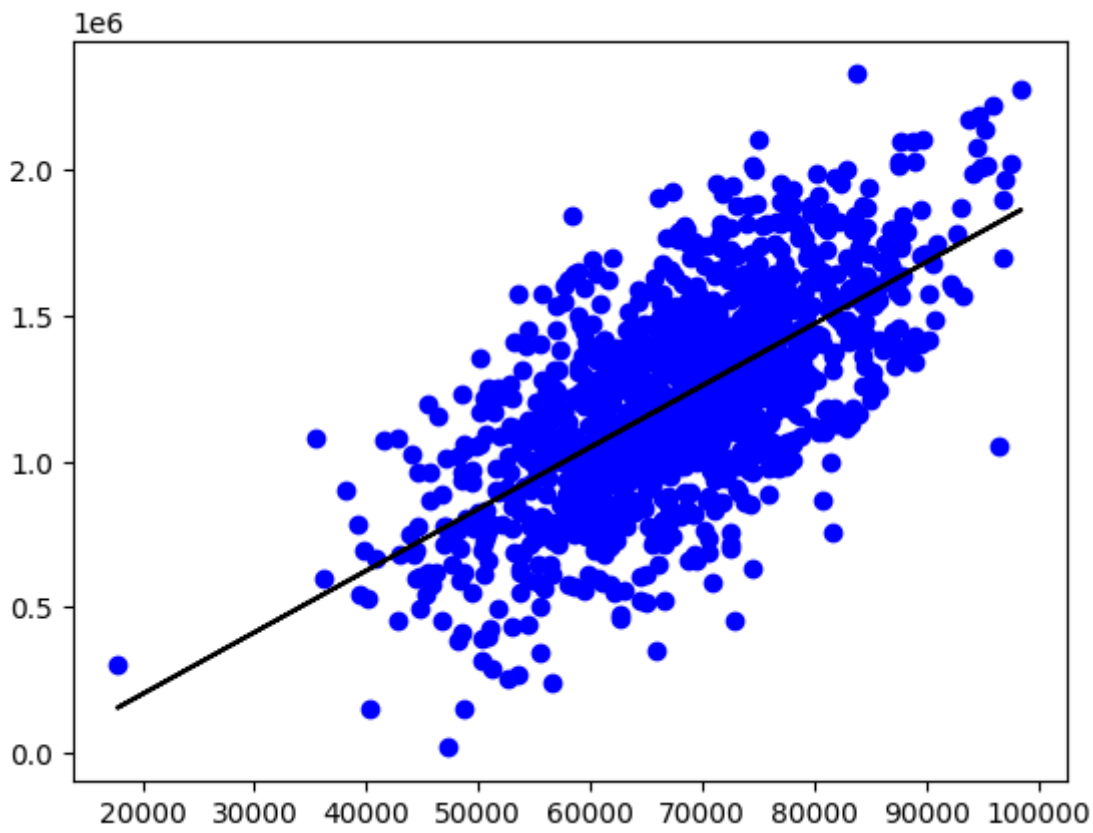
In [12]:
```
df.dropna()
```

Out[12]:

	avg	cost
0	79545.458574	1.059034e+06
1	79248.642455	1.505891e+06
2	61287.067179	1.058988e+06
3	63345.240046	1.260617e+06
4	59982.197226	6.309435e+05
...	...	...
4995	60567.944140	1.060194e+06
4996	78491.275435	1.482618e+06
4997	63390.686886	1.030730e+06
4998	68001.331235	1.198657e+06
4999	65510.581804	1.298950e+06

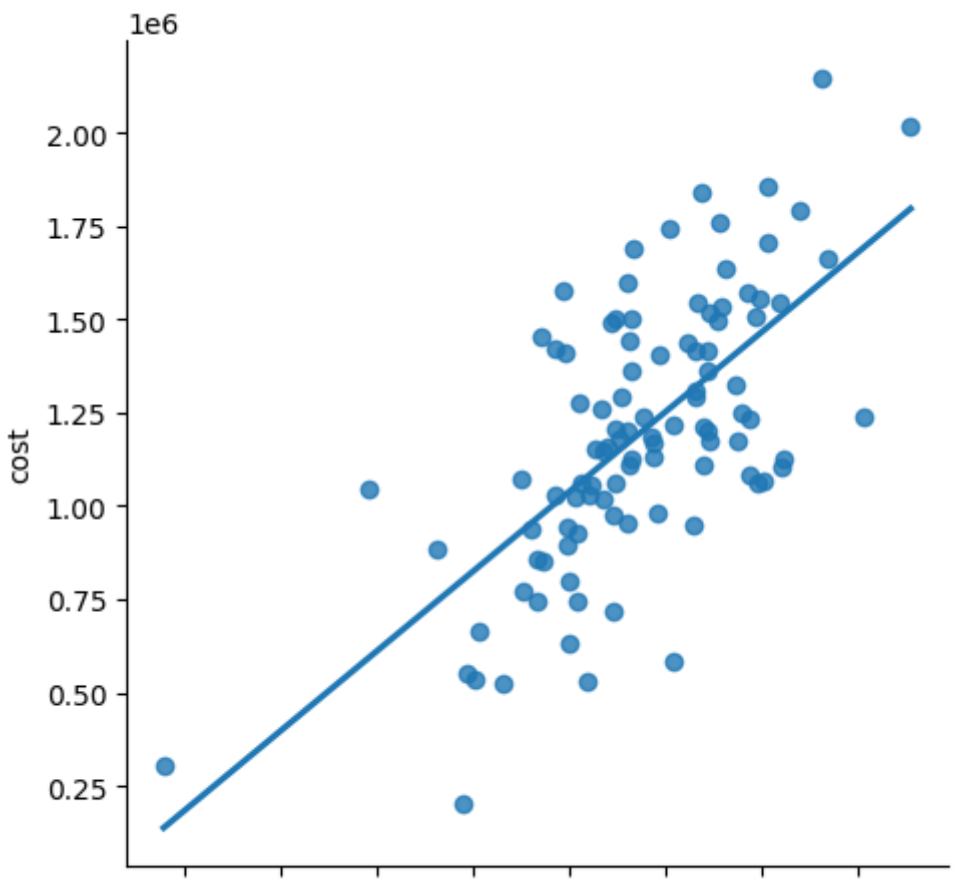
5000 rows × 2 columns

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()
```



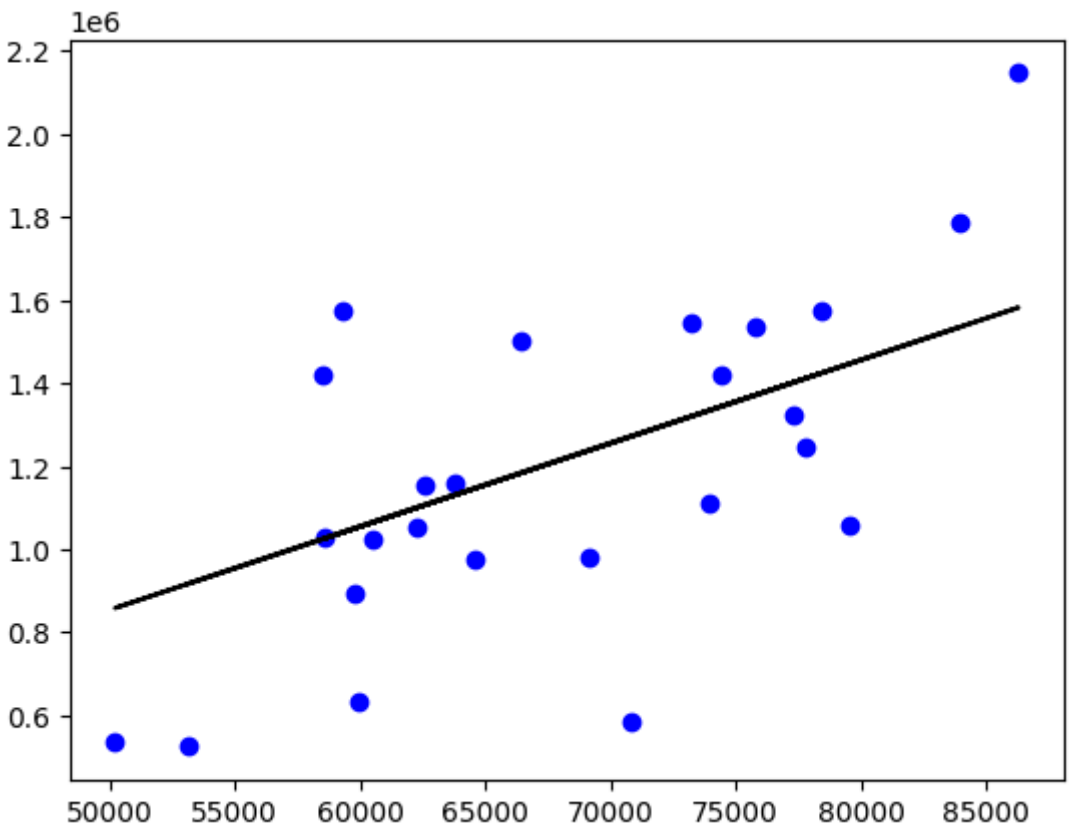
In [29]:
```
df100=df[:][:100]
sns.lmplot(x='avg',y='cost',data=df100,order=1,ci=None)
```

Out[29]: <seaborn.axisgrid.FacetGrid at 0x13a5d4feec0>



In [30]:
```
df100.fillna(method='ffill',inplace=True)
x = np.array(df100['avg']).reshape(-1,1)
y = np.array(df100['cost']).reshape(-1,1)
df100.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:  0.4043206804632161
```



In [31]:
```
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:  0.4043206804632161
```

Conclusion:

In this dataset having minimal amount of data so LinearRegression is same for both Normal data and Minimal amount of data

In [ ]:
```
1
```