In [18]:

```
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 [19]:

```
df=pd.read_csv(r"C:\Users\Prathyusha\Downloads\fiat500_VehicleSelection_Dataset.csv")
df
```

Out[19]:

	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 [20]:

```
df=df[['age_in_days','price']]
df.head(10)
```

Out[20]:

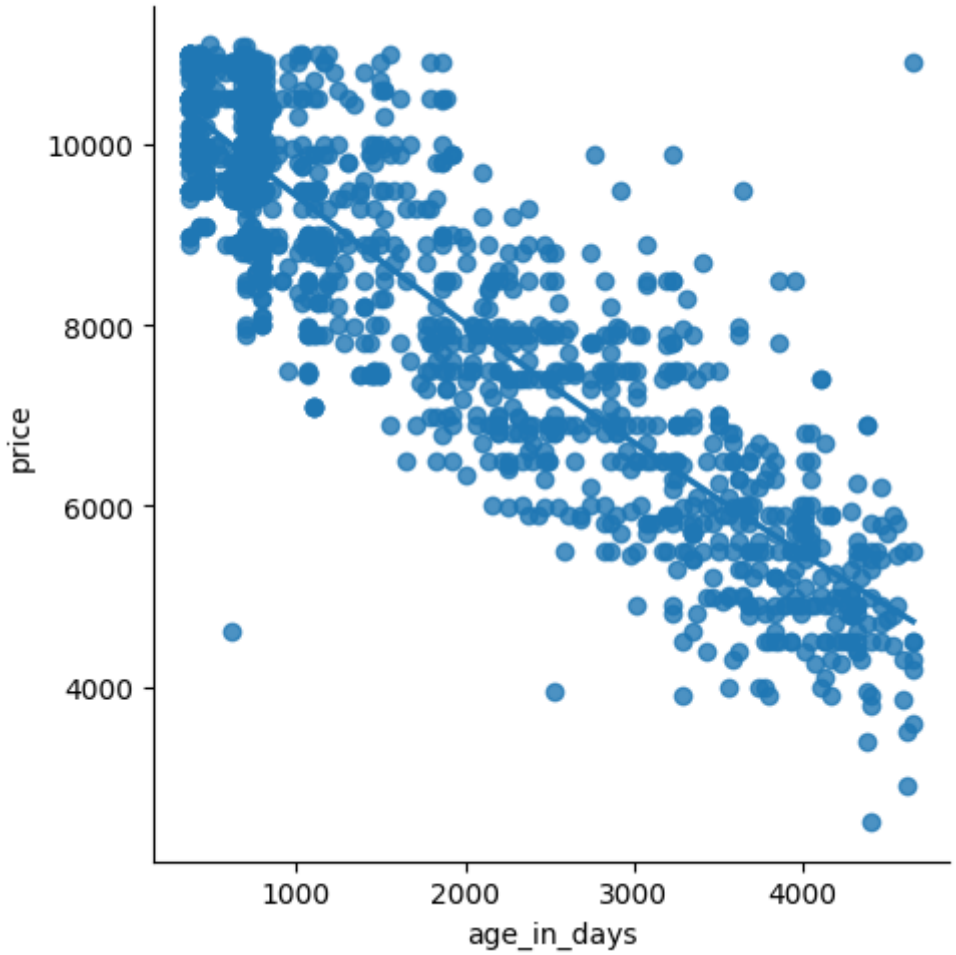
	age_in_days	price
0	882	8900
1	1186	8800
2	4658	4200
3	2739	6000
4	3074	5700
5	3623	7900
6	731	10750
7	1521	9190
8	4049	5600
9	3653	6000

In [21]:

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

Out[21]:

```
<seaborn.axisgrid.FacetGrid at 0x1b79e456c80>
```



In [22]:

```
df.describe()
```

Out[22]:

	age_in_days	price
count	1538.000000	1538.000000
mean	1650.980494	8576.003901
std	1289.522278	1939.958641
min	366.000000	2500.000000
25%	670.000000	7122.500000
50%	1035.000000	9000.000000
75%	2616.000000	10000.000000
max	4658.000000	11100.000000

In [23]:

```
df.info()
```

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

In [24]:

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

```
C:\Users\Prathyusha\AppData\Local\Temp\ipykernel_29560\1844562654.py:1: Se
ttingWithCopyWarning:
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 [25]:

```
x=np.array(df['age_in_days']).reshape(-1,1)
y=np.array(df['price']).reshape(-1,1)
```

In [26]:

```
df.dropna(inplace=True)
```

```
C:\Users\Prathyusha\AppData\Local\Temp\ipykernel_29560\1379821321.py:1: Se
ttingWithCopyWarning:
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 [27]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
```

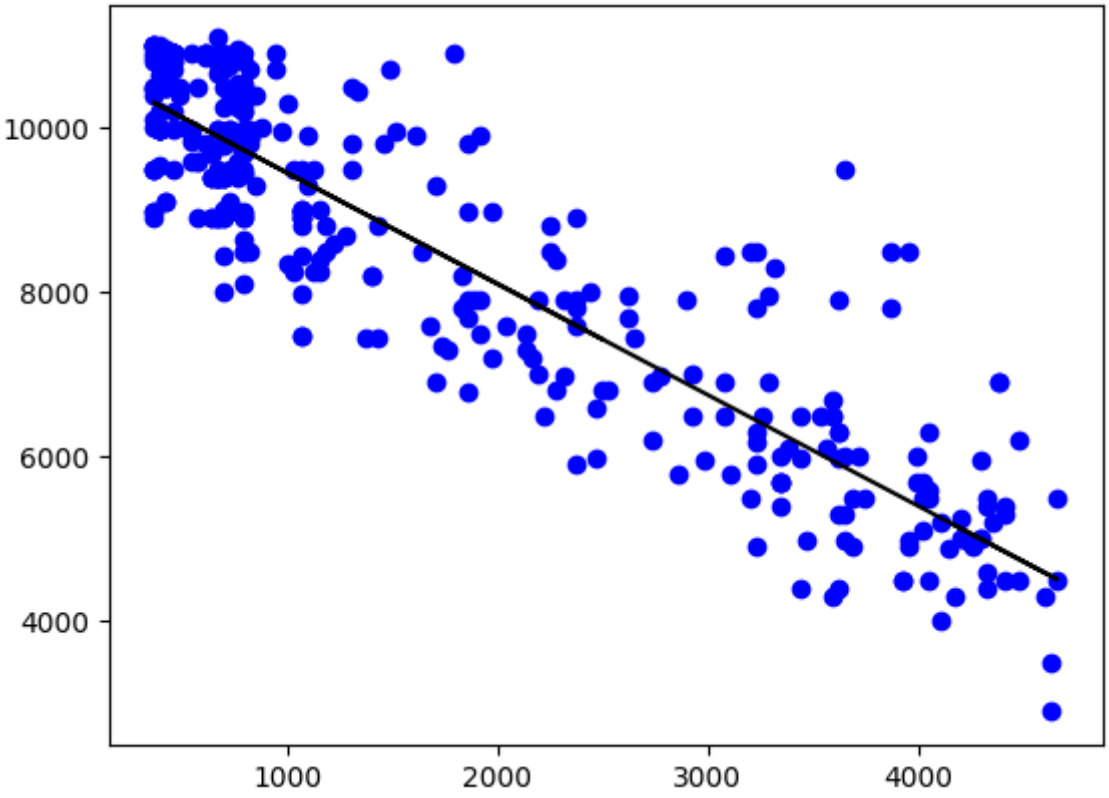
In [28]:

```
regr=LinearRegression()
regr.fit(x_train,y_train)
print(regr.score(x_test,y_test))
```

```
0.813993826673305
```

In [29]:

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



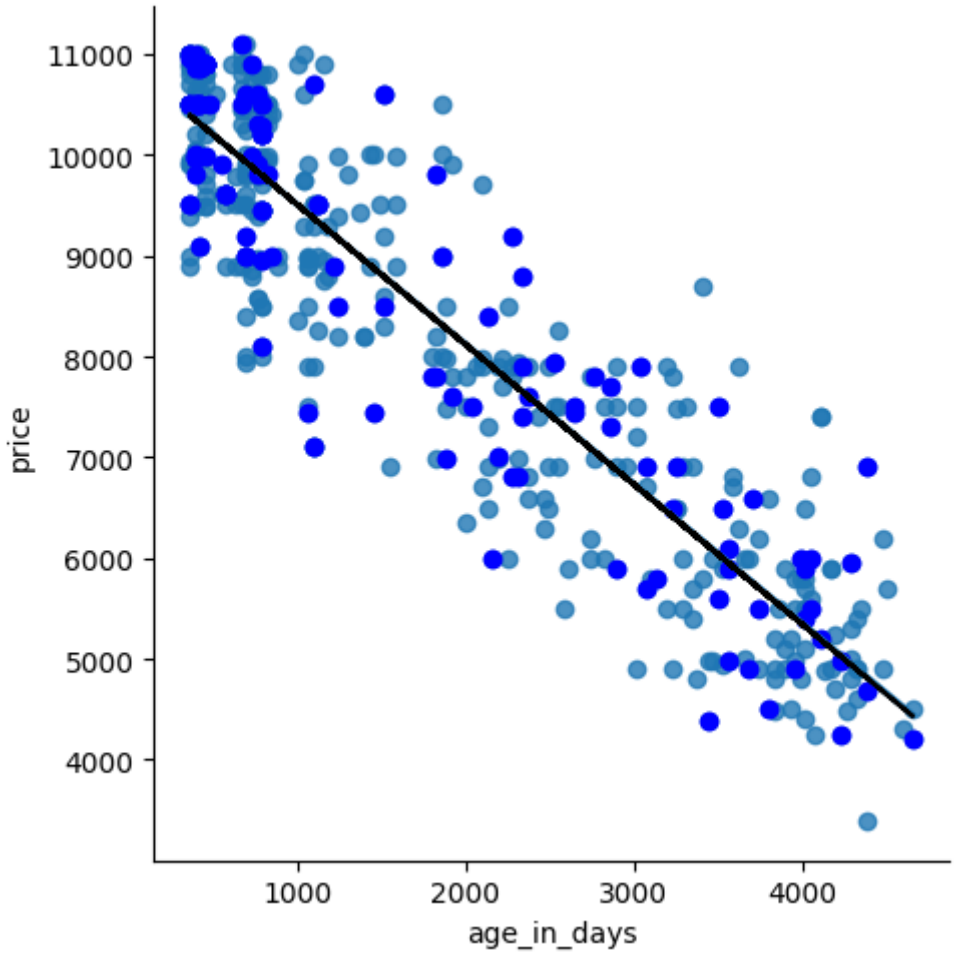
In [33]:

```
df500=df[:][:500]
```

In [34]:

```
sns.lmplot(x="age_in_days",y="price",data=df500,order=1,ci=None)
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['age_in_days']).reshape(-1,1)
y=np.array(df500['price']).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: 0.847275116420123
```



In [35]:

```
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: 0.847275116420123
```

In [16]:

```
coclusion:
    hence,this dataset is fit.
```

```
  Cell In[16], line 1
    coclusion:
              ^
SyntaxError: invalid syntax
```