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\teppa\Downloads\used_cars_data (1).csv")
df
```

Out[3]:

	S.No.	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_
0	0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	
1	1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	
2	2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	
3	3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	
4	4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Se
...	...	...	...	...	...	...	...	
7248	7248	Volkswagen Vento Diesel Trendline	Hyderabad	2011	89411	Diesel	Manual	
7249	7249	Volkswagen Polo GT TSI	Mumbai	2015	59000	Petrol	Automatic	
7250	7250	Nissan Micra Diesel XV	Kolkata	2012	28000	Diesel	Manual	
7251	7251	Volkswagen Polo GT TSI	Pune	2013	52262	Petrol	Automatic	
7252	7252	Mercedes-Benz E-Class 2009-2013 E 220 CDI Avan...	Kochi	2014	72443	Diesel	Automatic	

7253 rows × 14 columns

In [4]:
```
1 df=df[['Kilometers_Driven','Year']]
```

In [5]:
```
1 df.columns=['kil','year']
```

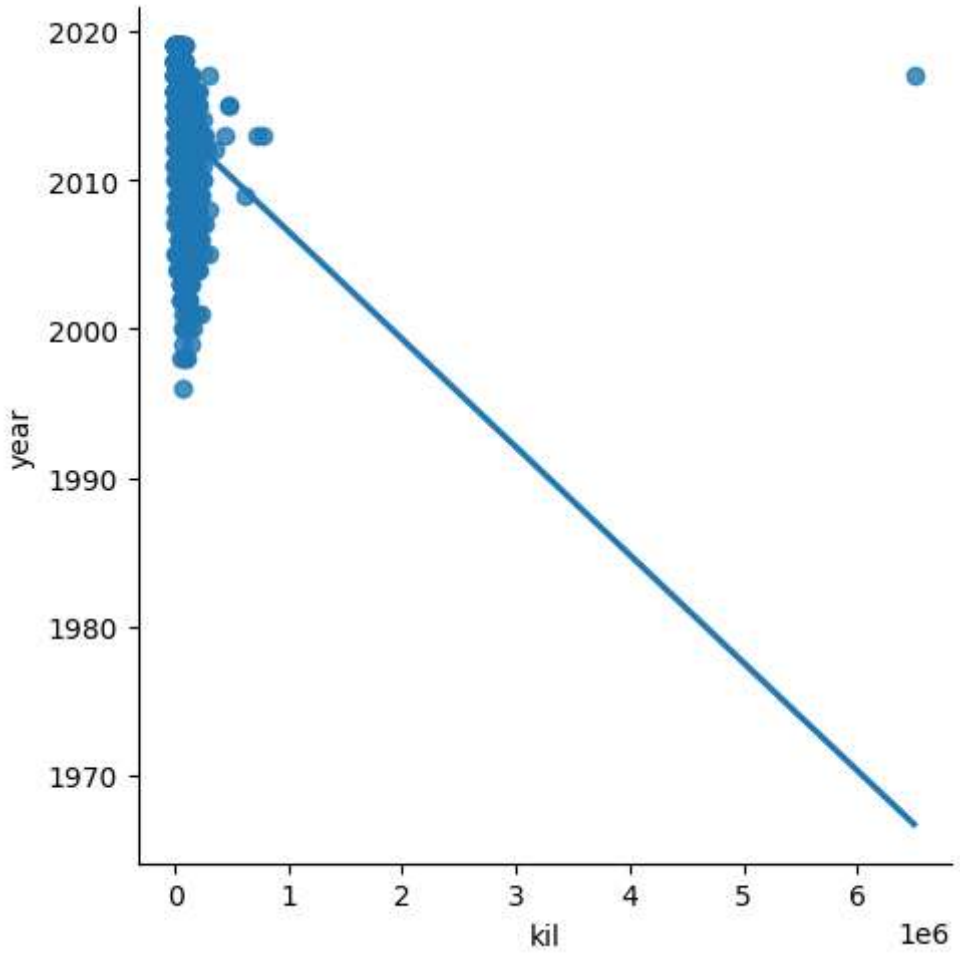
In [6]:
```
1 df.head(10)
```

Out[6]:

	kil	year
0	72000	2010
1	41000	2015
2	46000	2011
3	87000	2012
4	40670	2013
5	75000	2012
6	86999	2013
7	36000	2016
8	64430	2013
9	65932	2012

In [7]:
```
1 sns.lmplot(x="kil",y="year",data=df,order=1,ci=None)
```

Out[7]: <seaborn.axisgrid.FacetGrid at 0x18c8c71c1f0>



In [8]:
```
1 df.describe()
```

Out[8]:

	kil	year
count	7.253000e+03	7253.000000
mean	5.869906e+04	2013.365366
std	8.442772e+04	3.254421
min	1.710000e+02	1996.000000
25%	3.400000e+04	2011.000000
50%	5.341600e+04	2014.000000
75%	7.300000e+04	2016.000000
max	6.500000e+06	2019.000000

In [9]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7253 entries, 0 to 7252
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   kil     7253 non-null   int64
 1   year    7253 non-null   int64
dtypes: int64(2)
memory usage: 113.5 KB
```

In [10]:
```
1 df.fillna(method="ffill")
```

Out[10]:

	kil	year
0	72000	2010
1	41000	2015
2	46000	2011
3	87000	2012
4	40670	2013
...	...	...
7248	89411	2011
7249	59000	2015
7250	28000	2012
7251	52262	2013
7252	72443	2014

7253 rows × 2 columns

In [11]:
```
1 x=np.array(df["kil"]).reshape(-1,1)
2 y=np.array(df["year"]).reshape(-1,1)
```

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

In [13]:
```
df.fillna(method='ffill')
```

Out[13]:

	kil	year
0	72000	2010
1	41000	2015
2	46000	2011
3	87000	2012
4	40670	2013
...	...	...
7248	89411	2011
7249	59000	2015
7250	28000	2012
7251	52262	2013
7252	72443	2014

7253 rows × 2 columns

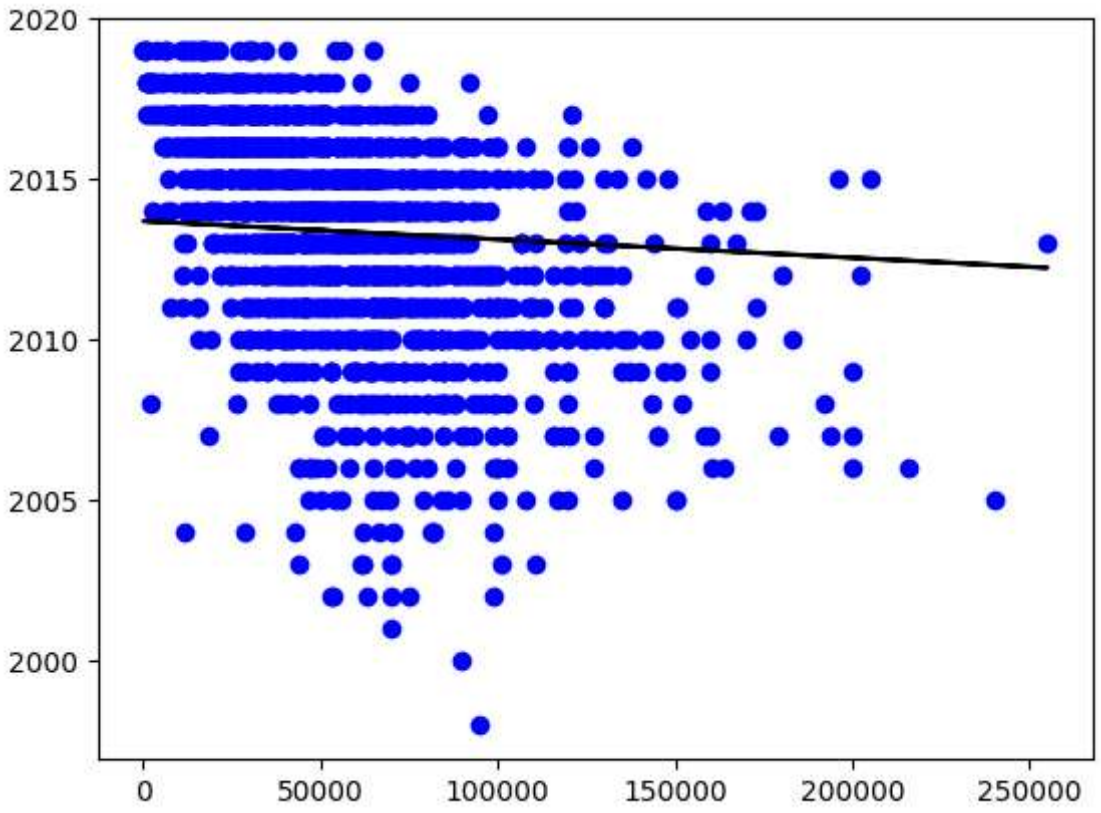
In [14]:
```
#df.dropna()
```

In [15]:
```
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))
```

```
0.051709057175647
```

In [16]:

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

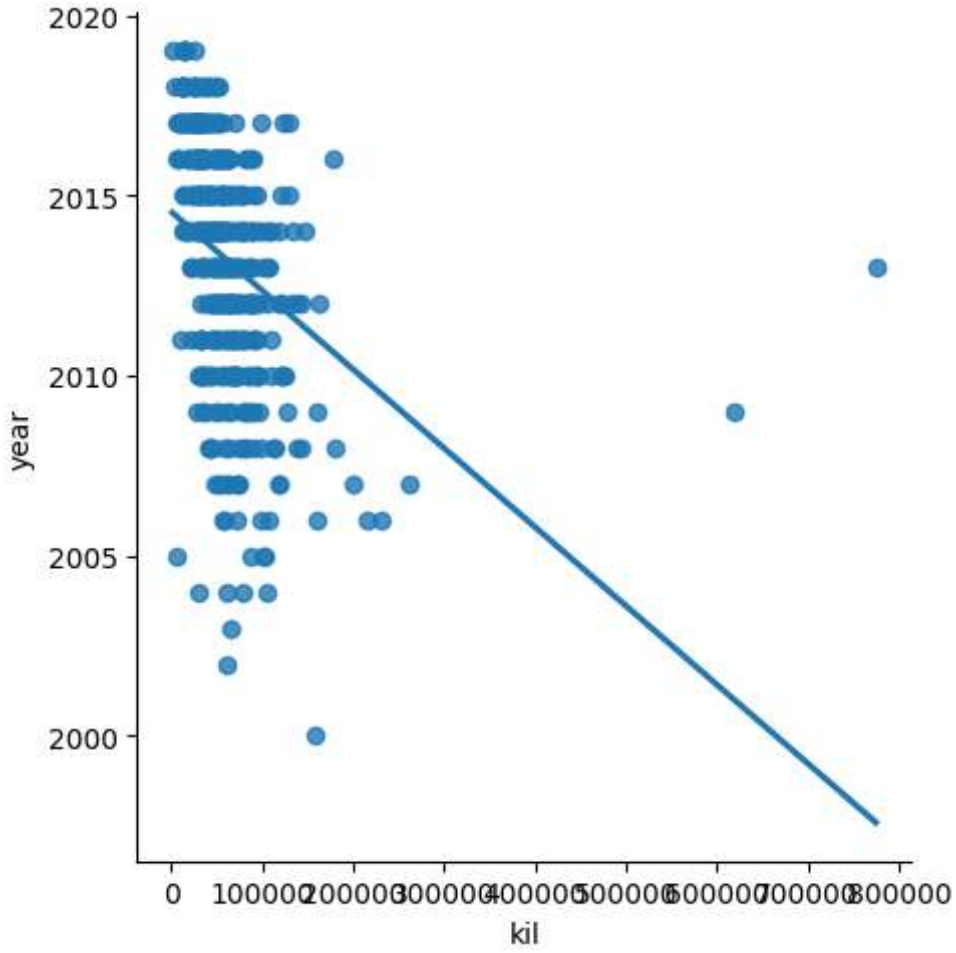


In [17]:

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

In [18]:
```
1 sns.lmplot(x="kil",y="year",data=df500,order=1,ci=None)
```

Out[18]: <seaborn.axisgrid.FacetGrid at 0x18c90cf0730>



In [19]:

```
df.fillna(method="ffill")
x=np.array(df500['kil']).reshape(-1,1)
y=np.array(df500['year']).reshape(-1,1)
df500.dropna()

```

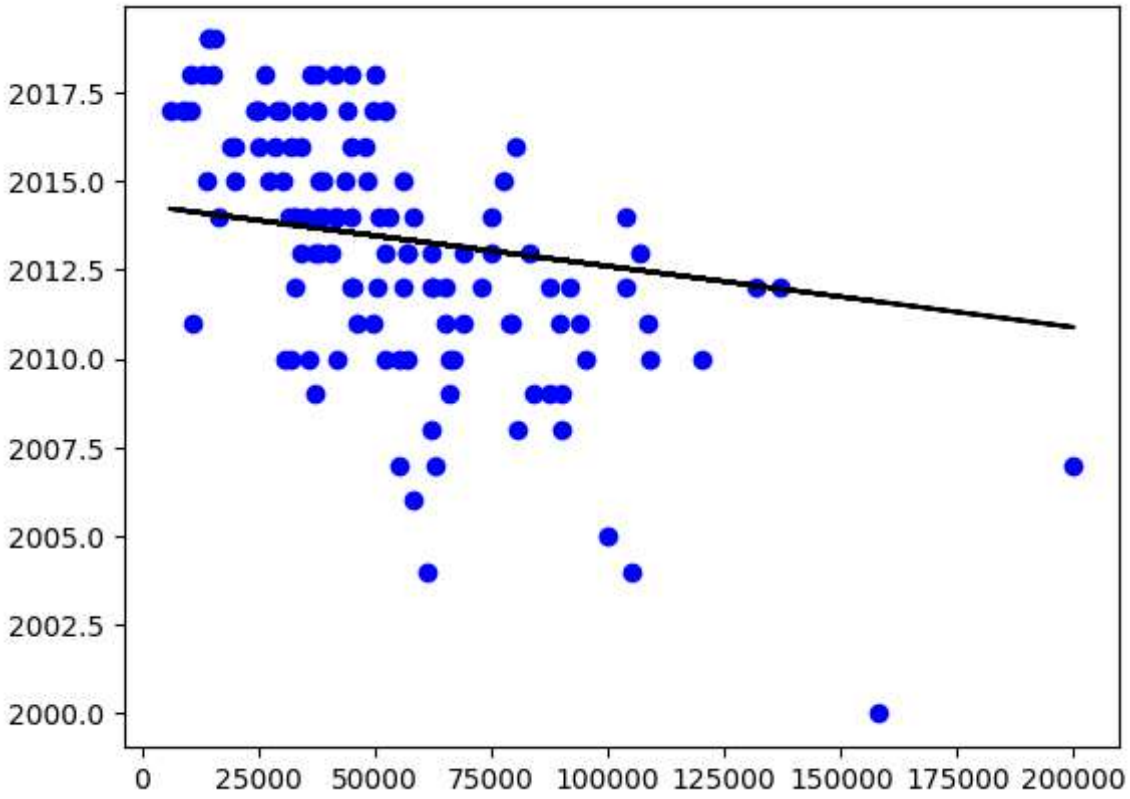
Out[19]:

	kil	year
0	72000	2010
1	41000	2015
2	46000	2011
3	87000	2012
4	40670	2013
...	...	...
495	72000	2006
496	79000	2015
497	29000	2004
498	66012	2013
499	35000	2014

500 rows × 2 columns

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("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 0.1668427773525476
```



In [21]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

In [22]:
```
model=LinearRegression()
model.fit(x_train,y_train)
```

Out[22]:
```
▾ LinearRegression

LinearRegression()
```

In [23]:
```
y_pred=model.predict(x_test)
r2=r2_score(y_test,y_pred)
print("Rescore:",r2)
```

```
Rescore: 0.1668427773525476
```

In [ ]:

```
1
```