In [1]:

```
#Step-1 Importing all the required 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 [3]:

```
#Step-2: Reading the Dataset

df=pd.read_csv(r"C:\Users\Mastan Reddy\Downloads\car.csv")
df
```

Out[3]:

	S.No.	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_T
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	Sec
...	...	...	...	...	...	...	...	
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	T
7252	7252	Mercedes-Benz E-Class 2009-2013 E 220 CDI Avan...	Kochi	2014	72443	Diesel	Automatic	

7253 rows × 14 columns

In [4]:

```
df = df[['Kilometers_Driven','Year']]
#Taking only selected two attributes from dataset
df.columns = ['kil','yr']
```

In [5]:
```
print('This Dataframe contains %d Rows and %d Columns'%(df.shape))
```

```
This Dataframe contains 7253 Rows and 2 Columns
```

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

Out[6]:

	kil	yr
0	72000	2010
1	41000	2015
2	46000	2011
3	87000	2012
4	40670	2013

In [7]:
```
df.tail()
```

Out[7]:

	kil	yr
7248	89411	2011
7249	59000	2015
7250	28000	2012
7251	52262	2013
7252	72443	2014

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

Out[8]:

	kil	yr
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]:

```
df.describe()
```

Out[9]:

	kil	yr
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 [10]:

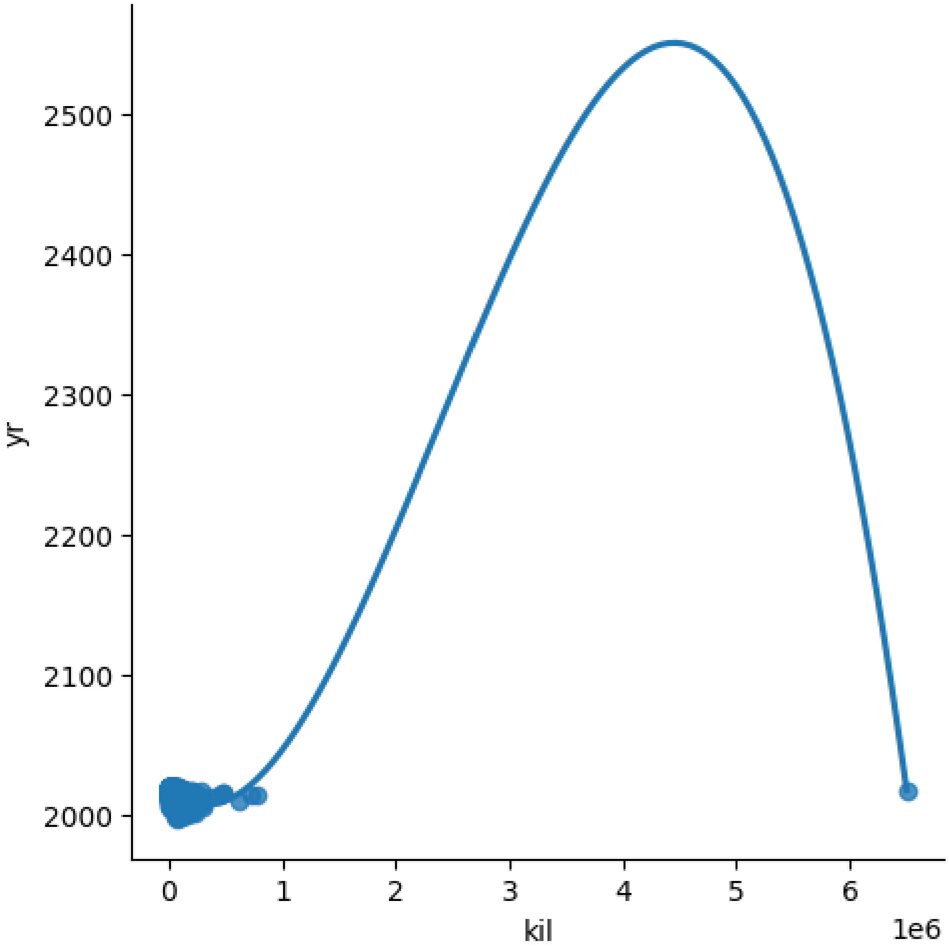
```
df.info()
```

```
<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   yr      7253 non-null   int64
dtypes: int64(2)
memory usage: 113.5 KB
```

In [11]:
```
#Step-3: Exploring the Data Scatter - plotting the data scatter

sns.lmplot(x="kil",y="yr", data = df, order = 3, ci = None)
```

Out[11]:
```
<seaborn.axisgrid.FacetGrid at 0xb34e32ed08>
```



In [12]:
```
#Step-4: Data cleaning - Eliminating NaN OR missing input numbers

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

```
C:\Reddy\Python37\lib\site-packages\pandas\core\frame.py:4468: SettingWithCop
yWarning:
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/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  downcast=downcast,
```

In [13]:

```
# Step-5: Training Our Model

X = np.array(df['kil']).reshape(-1, 1)

y = np.array(df['yr']).reshape(-1, 1)


#Seperating the data into independent and dependent variables and convert
#Now each dataset contains only one coloumn
```

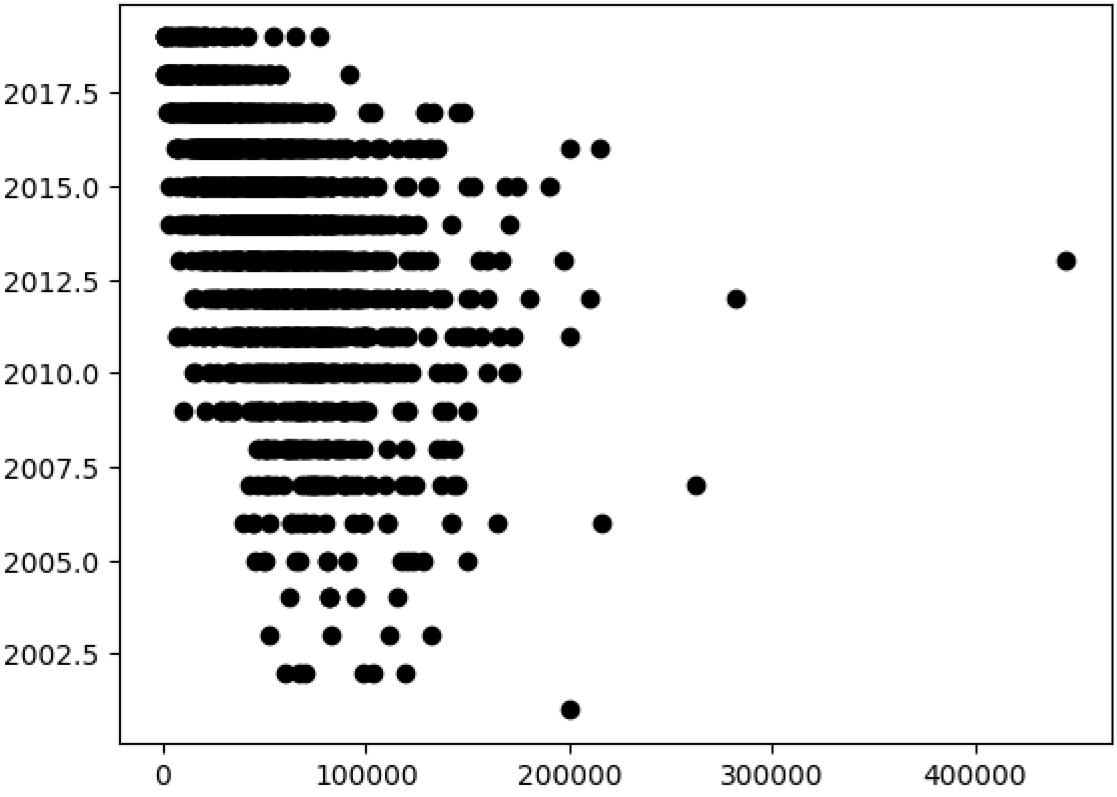
In [14]:

```
X_train,X_test,y_train,y_test = train_test_split(X, y, test_size = 0.25)
# Splitting the data into training data and test data
regr = LinearRegression()
regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

```
0.05255952411354059
```

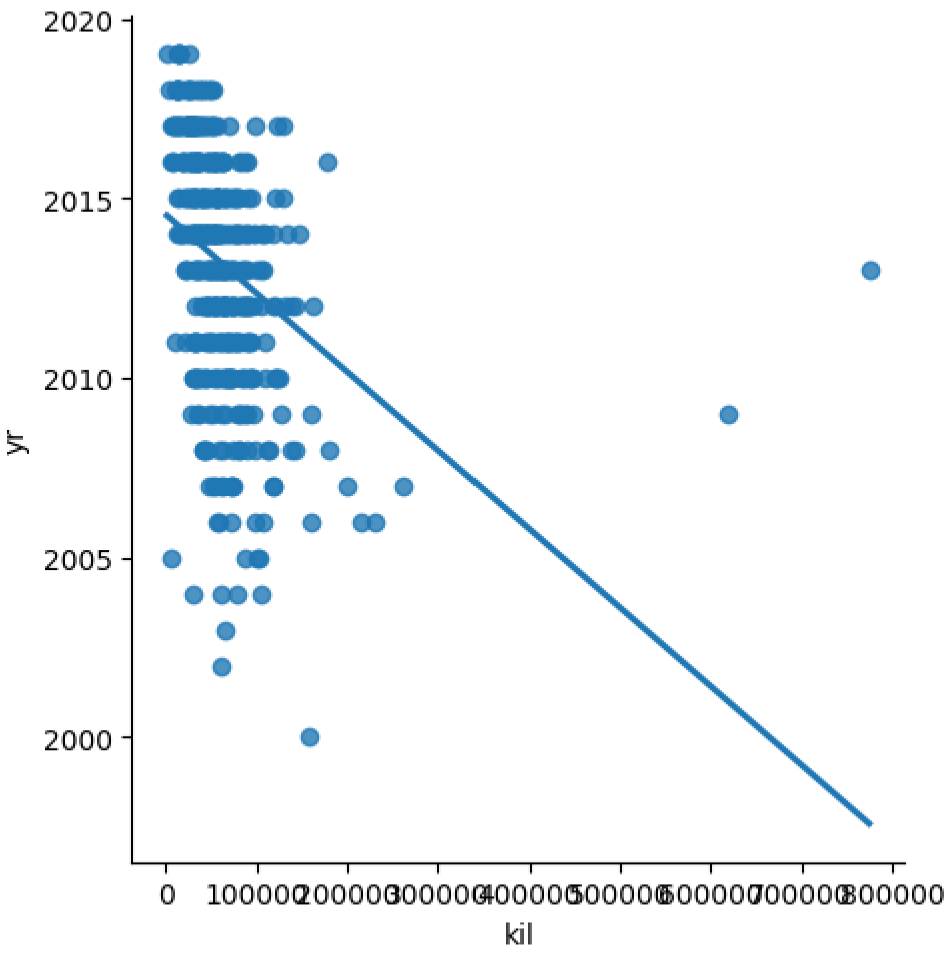
In [15]:

```
#step-6: Exploring Our Results
y_pred = regr.predict(X_test)
plt.scatter(X_test, y_test, color = 'k')
plt.show()
# Data scatter of predicted values
```



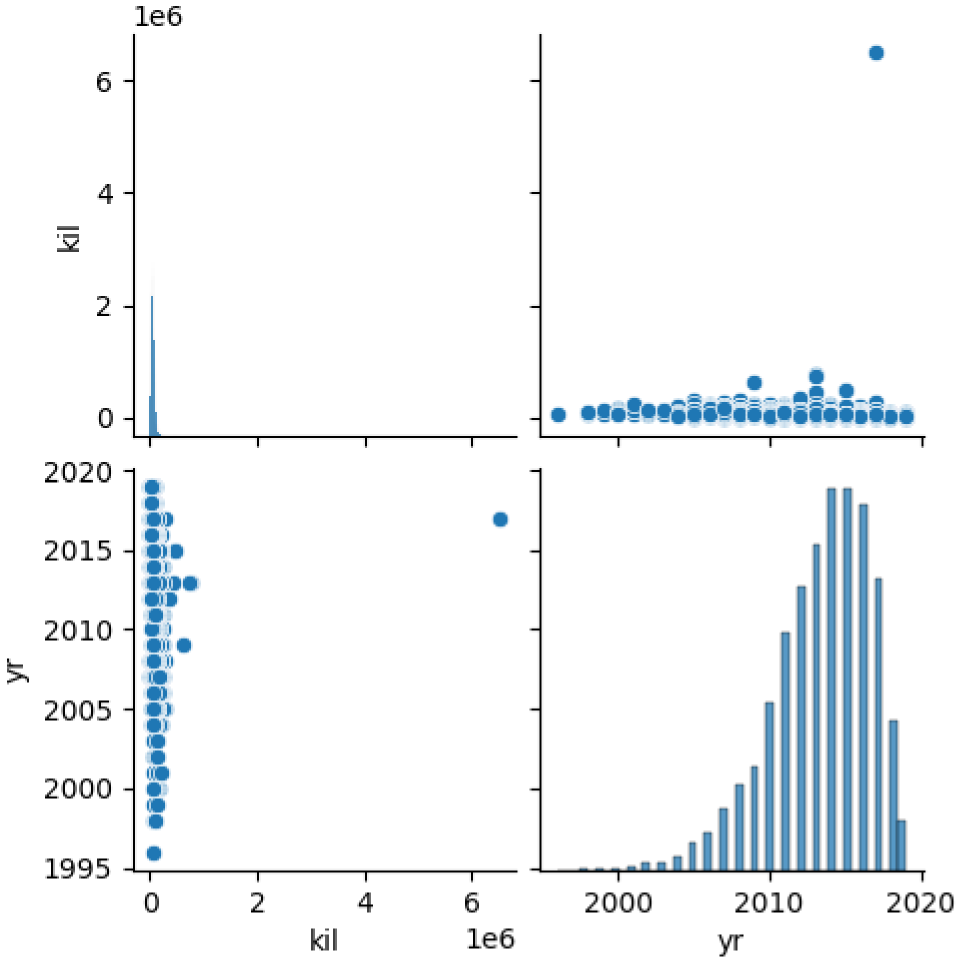
In [16]:
```
# Step-7: Working with a smaller Dataset
df500 = df[:][:500]
# Selecting the 1st 500 rows of teh data
sns.lmplot(x = "kil", y = "yr", data = df500, order = 1, ci = None)
```

Out[16]:
```
<seaborn.axisgrid.FacetGrid at 0xb358ba0e08>
```



```
In [17]: sns.pairplot(df)
```

```
Out[17]: <seaborn.axisgrid.PairGrid at 0xb358c907c8>
```



```
In [ ]:
```