A real estate agent want help to predict the house price for regions in Usa.he gave us the dataset to work on to use linear Regression model.Create a model that helps him to estimate

Data Collection

In [1]:

```
#import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [22]:

```
#import the dataset
data=pd.read_csv(r"C:\Users\user\Desktop\Vicky\1_fiat500_VehicleSelection_Dataset.csv")[0:500
```

In [23]:

```
#to display top 10 rows
data.head()
```

Out[23]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price	Unnamed: 9	U
0	1.0	lounge	51.0	882.0	25000.0	1.0	44.907242	8.611559868	8900	NaN	
1	2.0	pop	51.0	1186.0	32500.0	1.0	45.666359	12.24188995	8800	NaN	
2	3.0	sport	74.0	4658.0	142228.0	1.0	45.503300	11.41784	4200	NaN	
3	4.0	lounge	51.0	2739.0	160000.0	1.0	40.633171	17.63460922	6000	NaN	
4	5.0	pop	73.0	3074.0	106880.0	1.0	41.903221	12.49565029	5700	NaN	

In [24]:

```
#to display null values
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   ID               500 non-null    float64
 1   model            500 non-null    object 
 2   engine_power     500 non-null    float64
 3   age_in_days      500 non-null    float64
 4   km               500 non-null    float64
 5   previous_owners  500 non-null    float64
 6   lat              500 non-null    float64
 7   lon              500 non-null    object 
 8   price            500 non-null    object 
 9   Unnamed: 9       0 non-null      float64
 10  Unnamed: 10      0 non-null      object 
dtypes: float64(7), object(4)
memory usage: 43.1+ KB
```

In [25]:

```
#to display summary of statistics
data.describe()
```

Out[25]:

	ID	engine_power	age_in_days	km	previous_owners	lat	Unnamed: 9
count	500.000000	500.00000	500.000000	500.000000	500.00000	500.000000	0.0
mean	250.500000	51.90800	1677.516000	53279.784000	1.16000	43.664013	NaN
std	144.481833	4.03337	1339.277861	41893.569817	0.44135	2.139034	NaN
min	1.000000	51.00000	366.000000	1232.000000	1.00000	36.855839	NaN
25%	125.750000	51.00000	578.000000	18199.500000	1.00000	41.903221	NaN
50%	250.500000	51.00000	1066.000000	38000.000000	1.00000	44.508839	NaN
75%	375.250000	51.00000	2769.000000	81900.000000	1.00000	45.467960	NaN
max	500.000000	77.00000	4658.000000	188000.000000	4.00000	46.792019	NaN

In [26]:

```
#to display columns name
data.columns
```

Out[26]:

```
Index(['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owners',
       'lat', 'lon', 'price', 'Unnamed: 9', 'Unnamed: 10'],
      dtype='object')
```

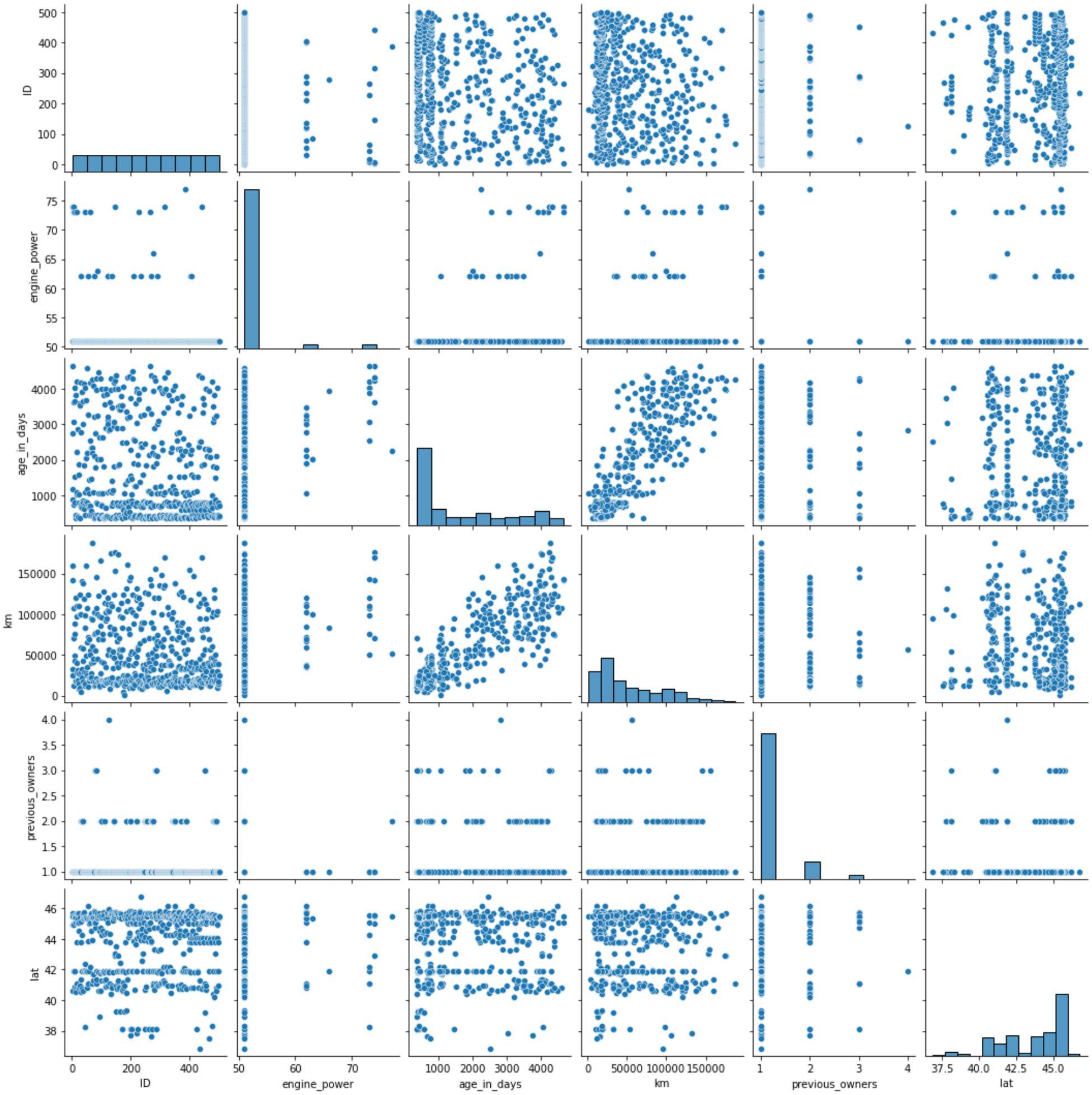
In [28]:

```
data1=data[['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owners',
       'lat', 'lon', 'price']]
```

EDA and Visualization

```
In [29]: sns.pairplot(data1)
```

```
Out[29]: <seaborn.axisgrid.PairGrid at 0x1760ca0d250>
```



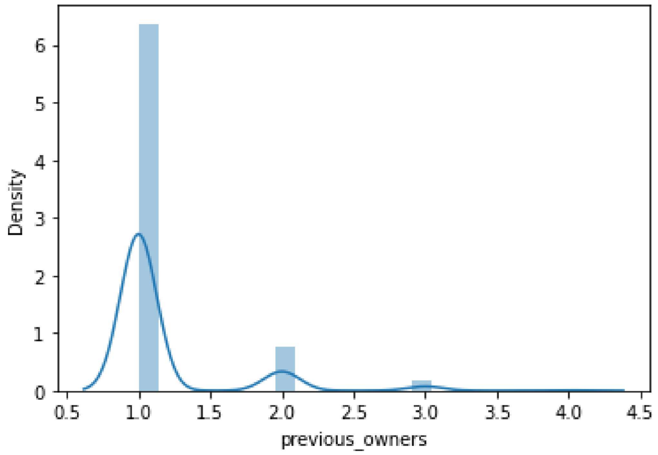
In [30]:

```
sns.distplot(data['previous_owners'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `di
stplot` is a deprecated function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar flexibility) or `histplot
` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[30]:

```
<AxesSubplot:xlabel='previous_owners', ylabel='Density'>
```



In [ ]:

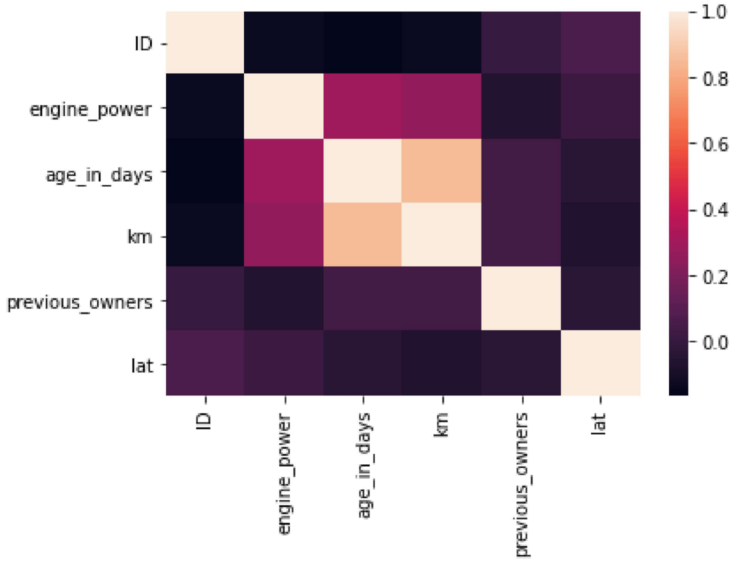
In [ ]:

In [31]:

```
sns.heatmap(data1.corr())
```

Out[31]:

```
<AxesSubplot:>
```



To train the model

we are going to train the linear regression model ;We need to split the two variable x and y where x in independent variable (input) and y is dependent of x(output) so we could ignore address columns as it is not requires for our model

In [103]:
```
x=data1[[ 'lat',  'price' ]]
y=data1['km']
```

In [104]:
```



#To split test and train data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.6)
```

In [105]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[105]:
```
LinearRegression()
```

In [106]:
```
lr.intercept_
```

Out[106]:
```
231360.35663553115
```

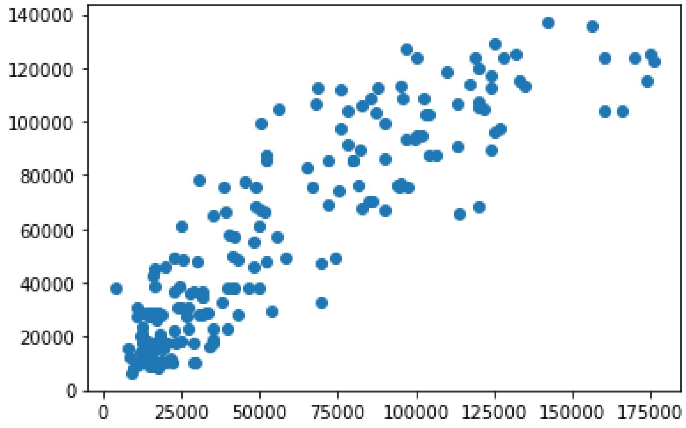
In [107]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=["Co-efficient"])
coeff
```

Out[107]:

	Co-efficient
lat	-322.678413
price	-18.980317

In [108]:
```
prediction = lr.predict(x_train)
plt.scatter(y_train,prediction)
```

Out[108]:
```
<matplotlib.collections.PathCollection at 0x1760e62d4f0>
```



In [90]:
```
lr.score(x_test,y_test)
```

Out[90]:
```
0.05380900324421822
```

In [ ]:

In [ ]: