Problem Statement

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 of what the house would sell for

Import libraries

In [1]:

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

Loading [MathJax]/extensions/Safe.js

In [2]:

```
# To import dataset
df=pd.read_csv('23_Vande Bharat.csv')
df
```

Loading [MathJax]/extensions/Safe.js

Out[2]:

	Sr. No.	Train Name	Train Number	Originating City	Originating Station	Terminal City	T
0	1	New Delhi - Varanasi Vande Bharat Express	22435/22436	Delhi	New Delhi	Varanasi	V
1	2	New Delhi - Shri Mata Vaishno Devi Katra Vande...	22439/22440	Delhi	New Delhi	Katra	Sl
2	3	Mumbai Central - Gandhinagar Capital Vande Bha...	20901/20902	Mumbai	Mumbai Central	Gandhinagar	Gan
3	4	New Delhi - Amb Andaura Vande Bharat Express	22447/22448	Delhi	New Delhi	Andaura	
4	5	MGR Chennai Central - Mysuru Vande Bharat Express	20607/20608	Chennai	Chennai Central	Mysuru	
5	6	Bilaspur - Nagpur Vande Bharat Express	20825/20826	Bilaspur, Chhattisgarh	Bilaspur Junction	Nagpur	
6	7	Howrah - New Jalpaiguri Vande Bharat Express	22301/22302	Kolkata	Howrah Junction	Siliguri	
7	8	Visakhapatnam - Secunderabad Vande Bharat Express	20833/20834	Visakhapatnam	Visakhapatnam Junction	Hyderabad	
8	9	Mumbai CSMT - Solapur Vande Bharat Express	22225/22226	Mumbai	Chhatrapati Shivaji Terminus	Solapur	
9	10	Mumbai CSMT - Sainagar Shirdi Vande Bharat Exp...	22223/22224	Mumbai	Chhatrapati Shivaji Terminus	Shirdi	
10	11	Rani Kamalapati (Habibganj) - Hazrat Nizamuddi...	20171/20172	Bhopal	Habibganj (Rani Kamalapati)	Delhi	Ha
11	12	Secunderabad - Tirupati Vande Bharat Express	20701/20702	Hyderabad	Secunderabad Junction	Tirupati	
12	13	MGR Chennai Central - Coimbatore Vande Bharat ...	20643/20644	Chennai	Chennai Central	Coimbatore	Coir
13	14	Delhi Cantonment - Ajmer Vande Bharat Express	20977/20978	Delhi	Delhi Cantonment	Ajmer	
14	15	Kasaragod - Thiruvananthapuram Vande Bharat Ex...	20633/20634	Kasaragod	Kasaragod	Thiruvananthapuram	Thiru
15	16	Howrah - Puri Vande Bharat Express	22895/22896	Kolkata	Howrah Junction	Puri	

Loading [MathJax]/extensions/Safe.js

	Sr. No.	Train Name	Train Number	Originating City	Originating Station	Terminal City	T
16	17	Anand Vihar Terminal - Dehradun Vande Bharat E...	22457/22458	Delhi	Anand Vihar Terminal	Dehradun	De
17	18	New Jalpaiguri - Guwahati Vande Bharat Express	22227/22228	Siliguri	New Jalpaiguri Junction	Guwahati	
18	19	Mumbai CSMT - Madgaon Vande Bharat Express	22229/22230	Mumbai	Chhatrapati Shivaji Terminus	Madgaon	M
19	19	Mumbai CSMT - Madgaon Vande Bharat Express	22229/22230	Mumbai	Chhatrapati Shivaji Terminus	Madgaon	M
20	20	Patna - Ranchi Vande Bharat Express	22349/22350	Patna	Patna Junction	Ranchi	
21	21	KSR Bengaluru - Dharwad Vande Bharat Express	20661/20662	Bangalore	Bangalore City	Hubbali - Dharwad	
22	22	Rani Kamalapati (Habibganj) - Jabalpur Vande B...	20173/20174	Bhopal	Habibganj (Rani Kamalapati)	Jabalpur	J
23	23	Indore - Bhopal Vande Bharat Express	20911/20912	Indore	Indore Junction	Bhopal	
24	24	Jodhpur - Sabarmati (Ahmedabad) Vande Bharat E...	12461/12462	Jodhpur	Jodhpur Junction	Ahmedabad	Sa
25	25	Gorakhpur - Lucknow Charbagh Vande Bharat Express	22549/22550	Gorakhpur	Gorakhpur Junction	Charbagh	Luc

In [3]:

```
# To display top 10 rows
df.head(10)
```

Out[3]:

	Sr. No.	Train Name	Train Number	Originating City	Originating Station	Terminal City	Terminal Station	O
0	1	New Delhi - Varanasi Vande Bharat Express	22435/22436	Delhi	New Delhi	Varanasi	Varanasi Junction	
1	2	New Delhi - Shri Mata Vaishno Devi Katra Vande...	22439/22440	Delhi	New Delhi	Katra	Shri Mata Vaishno Devi Katra	
2	3	Mumbai Central - Gandhinagar Capital Vande Bha...	20901/20902	Mumbai	Mumbai Central	Gandhinagar	Gandhinagar Capital	
3	4	New Delhi - Amb Andaura Vande Bharat Express	22447/22448	Delhi	New Delhi	Andaura	Amb Andaura	
4	5	MGR Chennai Central - Mysuru Vande Bharat Express	20607/20608	Chennai	Chennai Central	Mysuru	Mysore Junction	
5	6	Bilaspur - Nagpur Vande Bharat Express	20825/20826	Bilaspur, Chhattisgarh	Bilaspur Junction	Nagpur	Nagpur Junction	
6	7	Howrah - New Jalpaiguri Vande Bharat Express	22301/22302	Kolkata	Howrah Junction	Siliguri	New Jalpaiguri Junction	
7	8	Visakhapatnam - Secunderabad Vande Bharat Express	20833/20834	Visakhapatnam	Visakhapatnam Junction	Hyderabad	Secunderabad Junction	
8	9	Mumbai CSMT - Solapur Vande Bharat Express	22225/22226	Mumbai	Chhatrapati Shivaji Terminus	Solapur	Solapur	
9	10	Mumbai CSMT - Sainagar Shirdi Vande Bharat Exp...	22223/22224	Mumbai	Chhatrapati Shivaji Terminus	Shirdi	Sainagar Shirdi	

Data Cleaning and Pre-Processing

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26 entries, 0 to 25
Data columns (total 16 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   Sr. No.              26 non-null     int64 
 1   Train Name           26 non-null     object
 2   Train Number         26 non-null     object
 3   Originating City     26 non-null     object
 4   Originating Station  26 non-null     object
 5   Terminal City        26 non-null     object
 6   Terminal Station     26 non-null     object
 7   Operator             26 non-null     object
 8   No. of Cars          26 non-null     int64 
 9   Frequency            26 non-null     object
 10  Distance             26 non-null     object
 11  Travel Time          26 non-null     object
 12  Speed                26 non-null     object
 13  Average Speed        26 non-null     object
 14  Inauguration         26 non-null     object
 15  Average occupancy    26 non-null     object
dtypes: int64(2), object(14)
memory usage: 3.4+ KB
```

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

Out[5]:

	Sr. No.	No. of Cars
count	26.000000	26.000000
mean	13.230769	12.923077
std	7.306478	3.969112
min	1.000000	8.000000
25%	7.250000	8.000000
50%	13.500000	16.000000
75%	19.000000	16.000000
max	25.000000	16.000000

In [6]:
```
df.columns
```

Out[6]:
```
Index(['Sr. No.', 'Train Name', 'Train Number', 'Originating City',
       'Originating Station', 'Terminal City', 'Terminal Station', 'Operato
r',
       'No. of Cars', 'Frequency', 'Distance', 'Travel Time', 'Speed',
       'Average Speed', 'Inauguration', 'Average occupancy'],
      dtype='object')
```

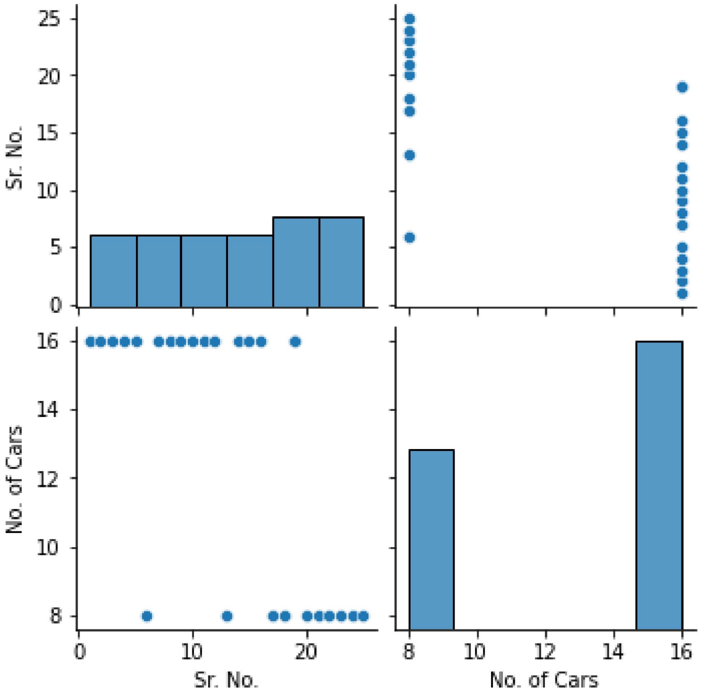
In [7]:
```
a = df.dropna(axis='columns')
a.columns
```

Out[7]:
```
Index(['Sr. No.', 'Train Name', 'Train Number', 'Originating City',
       'Originating Station', 'Terminal City', 'Terminal Station', 'Operato
r',
       'No. of Cars', 'Frequency', 'Distance', 'Travel Time', 'Speed',
       'Average Speed', 'Inauguration', 'Average occupancy'],
      dtype='object')
```

EDA and Visualization

In [8]:
```
sns.pairplot(a)
```

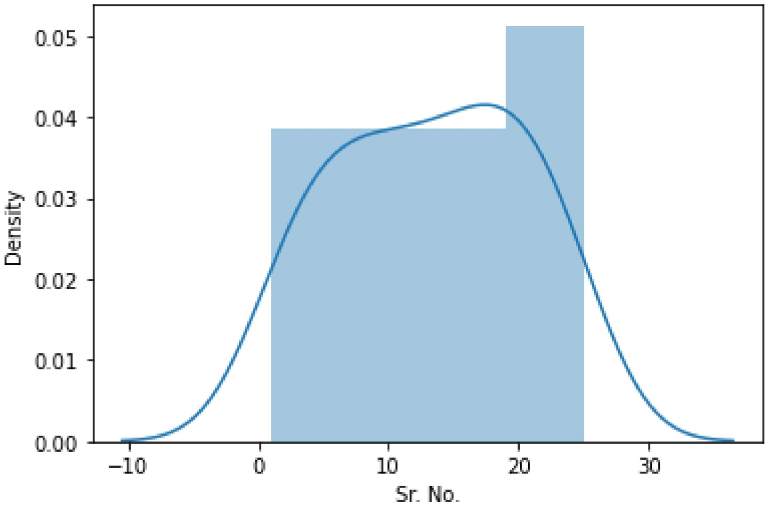
Out[8]:
```
<seaborn.axisgrid.PairGrid at 0x2ea6aad3490>
```



In [9]:
```
sns.distplot(a['Sr. No.'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: Fut
ureWarning: `distplot` is a deprecated function and will be removed in a futu
re version. Please adapt your code to use either `displot` (a figure-level fu
nction with similar flexibility) or `histplot` (an axes-level function for hi
stograms).
  warnings.warn(msg, FutureWarning)
```

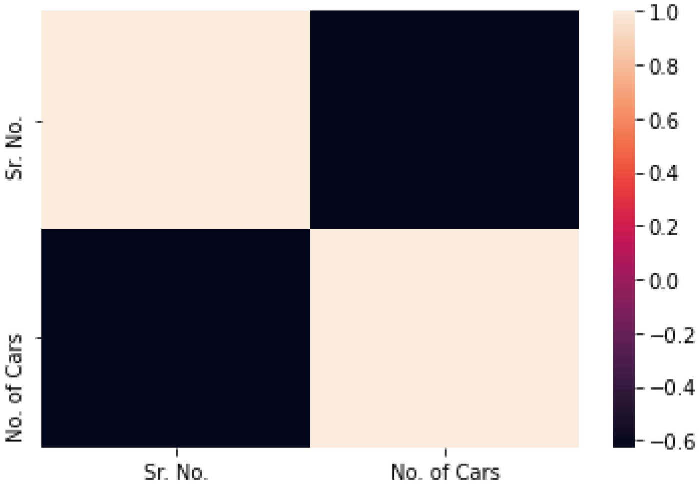
Out[9]:
```
<AxesSubplot:xlabel='Sr. No.', ylabel='Density'>
```



In [10]:
```
a1=a[['Sr. No.','No. of Cars']]
```

In [11]:
```
sns.heatmap(a1.corr())
```

Out[11]:
```
<AxesSubplot:>
```



To Train the Model - Model Building

We are going to train Linear Regression model;We need to split out data into two variables x and y where x is independent variable (input) and y is dependent on x(output). We could ignore address column as it is not required for our model.

In [12]:
```
x=a1[['No. of Cars']]
y=a1['Sr. No.']
```

To split my dataset into training and test data

In [13]:
```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [14]:
```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[14]:
```
LinearRegression()
```

In [15]:
```
print(lr.intercept_)
```

```
21.846153846153843
```

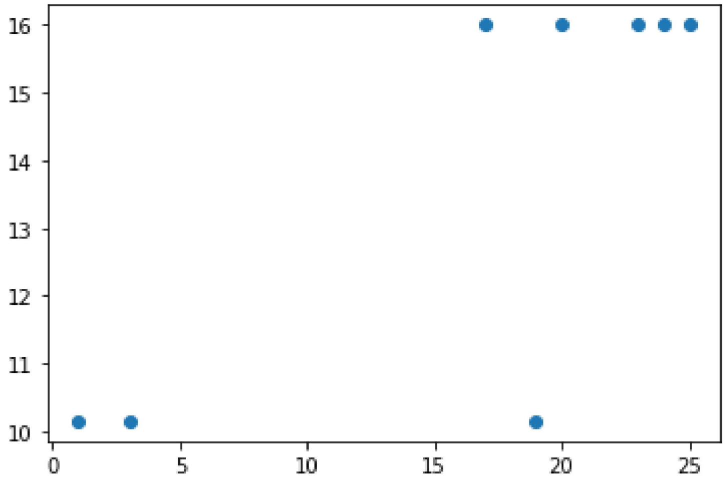
In [16]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[16]:

	Co-efficient
No. of Cars	-0.730769

In [17]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[17]:
```
<matplotlib.collections.PathCollection at 0x2ea6cd68b80>
```



```
In [18]: print(lr.score(x_test,y_test))

         0.3068221371388792
```

```
In [19]: from sklearn.linear_model import Ridge,Lasso
```

```
In [20]: rr=Ridge(alpha=10)
         rr.fit(x_train,y_train)

Out[20]: Ridge(alpha=10)
```

```
In [21]: 
         rr.score(x_train,y_train)

Out[21]: 0.20564883984761995
```

```
In [22]: rr.score(x_test,y_test)

Out[22]: 0.28831116213394836
```

```
In [23]: rr.score(x_test,y_test)

Out[23]: 0.28831116213394836
```

```
In [24]: la=Lasso(alpha=10)
         la.fit(x_train,y_train)

Out[24]: Lasso(alpha=10)
```

```
In [25]: la.score(x_test,y_test)

Out[25]: -0.291495198902606
```

```
In [26]: from sklearn.linear_model import ElasticNet
         en = ElasticNet()
         en.fit(x_train,y_train)

Out[26]: ElasticNet()
```

```
In [27]: print(en.coef_)

         [-0.66589542]
```

```
In [28]: print(en.intercept_)

         20.952336881073578
```

```
In [29]: print(en.predict(x_test))

         [15.62517353 15.62517353 15.62517353 15.62517353 10.29801018 15.62517353
          10.29801018 10.29801018]
```

Loading [MathJax]/extensions/Safe.js

In [30]:
```
print(en.score(x_test,y_test))
```

```
0.26653434255727493
```

Evaluation Metrics

In [31]:
```
from sklearn import metrics
print("Mean Absolytre Error:",metrics.mean_absolute_error(y_test,prediction))
print("Mean Squared Error:",metrics.mean_squared_error(y_test,prediction))
print("Root Mean Squared Error:",np.sqrt(metrics.mean_squared_error(y_test,pre
```

```
Mean Absolytre Error: 6.769230769230077
Mean Squared Error: 53.028106508875744
Root Mean Squared Error: 7.282039996379843
```

In [32]:
```

import pickle
```

In [35]:
```
filename='prediction5'
pickle.dump(lr,open(filename,'wb'))
```

In [ ]:

Loading [MathJax]/extensions/Safe.js