kaviyadevi 20106064

In [1]:
```
#to import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
#to import dataset
data1=pd.read_csv(r"C:\Users\user\Downloads\23_Vande Bharat - 23_Vande Bharat.csv
data1
```

Out[2]:

	Sr. No.	Train Name	Train Number	Originating City	Originating Station	Terminal City	Ter
0	1	New Delhi - Varanasi Vande Bharat Express	22435/22436	Delhi	New Delhi	Varanasi	Vara
1	2	New Delhi - Shri Mata Vaishno Devi Katra Vande...	22439/22440	Delhi	New Delhi	Katra	Shri
2	3	Mumbai Central - Gandhinagar Capital Vande Bha...	20901/20902	Mumbai	Mumbai Central	Gandhinagar	Gandh
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	My
5	6	Bilaspur - Nagpur Vande Bharat Express	20825/20826	Bilaspur, Chhattisgarh	Bilaspur Junction	Nagpur	Na
6	7	Howrah - New Jalpaiguri Vande Bharat Express	22301/22302	Kolkata	Howrah Junction	Siliguri	N
7	8	Visakhapatnam - Secunderabad Vande Bharat Express	20833/20834	Visakhapatnam	Visakhapatnam Junction	Hyderabad	S
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	S
10	11	Rani Kamalapati (Habibganj) - Hazrat Nizamuddi...	20171/20172	Bhopal	Habibganj (Rani Kamalapati)	Delhi	Hazra
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	Coimb
13	14	Delhi Cantonment - Ajmer Vande Bharat Express	20977/20978	Delhi	Delhi Cantonment	Ajmer	A

	Sr. No.	Train Name	Train Number	Originating City	Originating Station	Terminal City	Ter
14	15	Kasaragod - Thiruvananthapuram Vande Bharat Ex...	20633/20634	Kasaragod	Kasaragod	Thiruvananthapuram	Thiruva
15	16	Howrah - Puri Vande Bharat Express	22895/22896	Kolkata	Howrah Junction	Puri	
16	17	Anand Vihar Terminal - Dehradun Vande Bharat E...	22457/22458	Delhi	Anand Vihar Terminal	Dehradun	Dehra
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	Mad
19	19	Mumbai CSMT - Madgaon Vande Bharat Express	22229/22230	Mumbai	Chhatrapati Shivaji Terminus	Madgaon	Mad
20	20	Patna - Ranchi Vande Bharat Express	22349/22350	Patna	Patna Junction	Ranchi	Ra
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	Jab
23	23	Indore - Bhopal Vande Bharat Express	20911/20912	Indore	Indore Junction	Bhopal	Bh
24	24	Jodhpur - Sabarmati (Ahmedabad) Vande Bharat E...	12461/12462	Jodhpur	Jodhpur Junction	Ahmedabad	Saba
25	25	Gorakhpur - Lucknow Charbagh Vande Bharat Express	22549/22550	Gorakhpur	Gorakhpur Junction	Charbagh	Luckr

In [3]:
```
#to display top 5 rows
data=data1.head()
data
```

Out[3]:

	Sr. No.	Train Name	Train Number	Originating City	Originating Station	Terminal City	Terminal Station	Operator	N o Car
0	1	New Delhi - Varanasi Vande Bharat Express	22435/22436	Delhi	New Delhi	Varanasi	Varanasi Junction	NR	1
1	2	New Delhi - Shri Mata Vaishno Devi Katra Vande...	22439/22440	Delhi	New Delhi	Katra	Shri Mata Vaishno Devi Katra	NR	1
2	3	Mumbai Central - Gandhinagar Capital Vande Bha...	20901/20902	Mumbai	Mumbai Central	Gandhinagar	Gandhinagar Capital	WR	1
3	4	New Delhi - Amb Andaura Vande Bharat Express	22447/22448	Delhi	New Delhi	Andaura	Amb Andaura	NR	1
4	5	MGR Chennai Central - Mysuru Vande Bharat Express	20607/20608	Chennai	Chennai Central	Mysuru	Mysore Junction	SR	1

DATA CLEANING AND PREPROCESSING

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype 
---  ------                --------------  ----- 
 0   Sr. No.               5 non-null      int64 
 1   Train Name            5 non-null      object
 2   Train Number          5 non-null      object
 3   Originating City      5 non-null      object
 4   Originating Station   5 non-null      object
 5   Terminal City         5 non-null      object
 6   Terminal Station      5 non-null      object
 7   Operator              5 non-null      object
 8   No. of Cars           5 non-null      int64 
 9   Frequency             5 non-null      object
 10  Distance              5 non-null      object
 11  Travel Time           5 non-null      object
 12  Speed                 5 non-null      object
 13  Average Speed         5 non-null      object
 14  Inauguration          5 non-null      object
 15  Average occupancy     5 non-null      object
dtypes: int64(2), object(14)
memory usage: 768.0+ bytes
```

In [5]:
```
#to display summary of statistics
data.describe()
```

Out[5]:

	Sr. No.	No. of Cars
count	5.000000	5.0
mean	3.000000	16.0
std	1.581139	0.0
min	1.000000	16.0
25%	2.000000	16.0
50%	3.000000	16.0
75%	4.000000	16.0
max	5.000000	16.0

In [6]:
```
data.isnull()
```

Out[6]:

	Sr. No.	Train Name	Train Number	Originating City	Originating Station	Terminal City	Terminal Station	Operator	No. of Cars	Frequency
0	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False

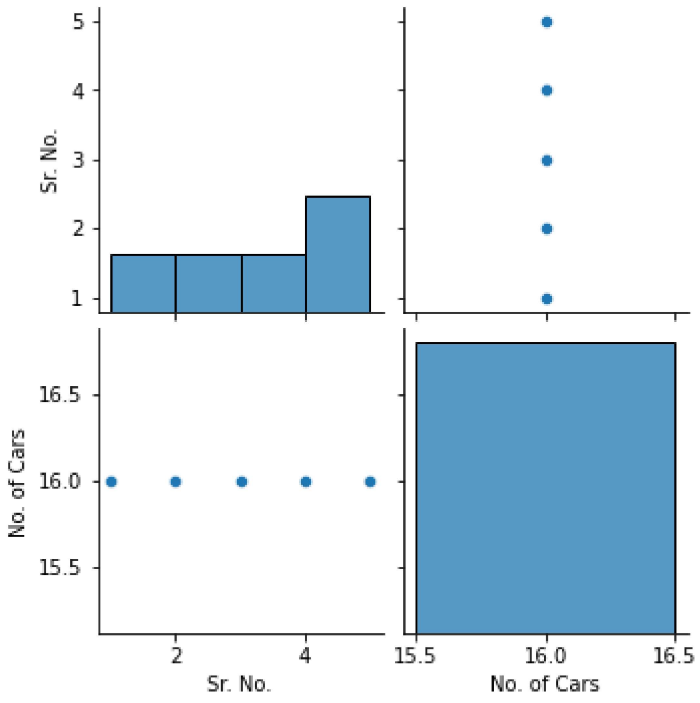
In [7]:
```
#to display the column heading
data.columns
```

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

EDA and DATA VISUALIZATION

In [12]:
```
sns.pairplot(data)
```

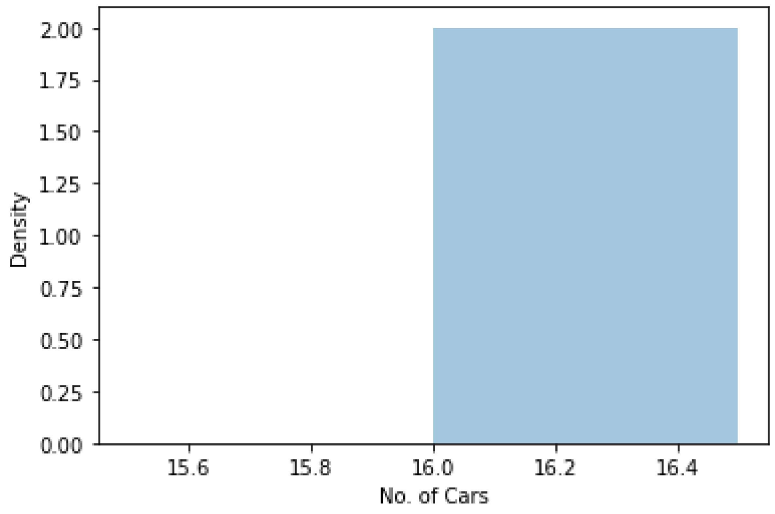
Out[12]:
```
<seaborn.axisgrid.PairGrid at 0x2a12f0dedc0>
```



In [14]:
```
sns.distplot(data['No. of Cars'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:306: UserWa
rning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
```

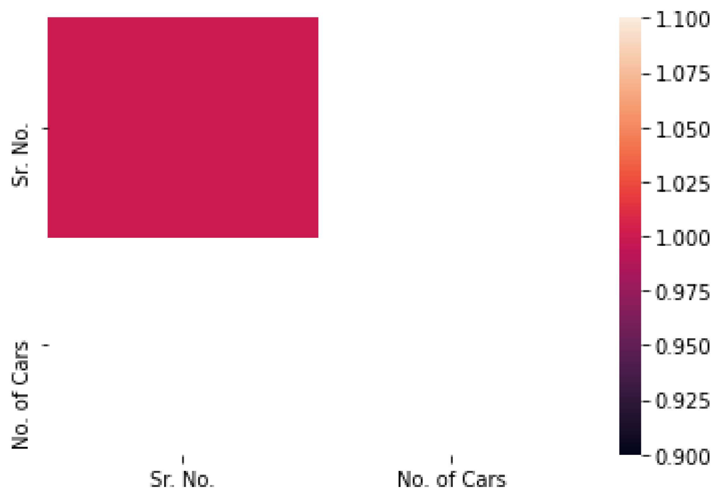
Out[14]:
```
<AxesSubplot:xlabel='No. of Cars', ylabel='Density'>
```



In [8]:
```
df=data[['Sr. No.', 'Train Name', 'Train Number', 'Originating City',
       'Originating Station', 'Terminal City', 'Terminal Station', 'Operator',
       'No. of Cars', 'Frequency', 'Distance', 'Travel Time', 'Speed',
       'Average Speed', 'Inauguration', 'Average occupancy']]
```

In [15]:
```
sns.heatmap(df.corr())
```

Out[15]:
```
<AxesSubplot:>
```



TO TRAIN MODEL

In [20]:
```
x=df[['Sr. No.']]
y=df['No. of Cars']
```

In [21]:
```
#to split my dataset into trainning and test

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 [22]:
```
from sklearn.linear_model import LinearRegression

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

Out[22]:
```
LinearRegression()
```

In [23]:
```
#to find intercept
print(lr.intercept_)
```

```
16.0
```

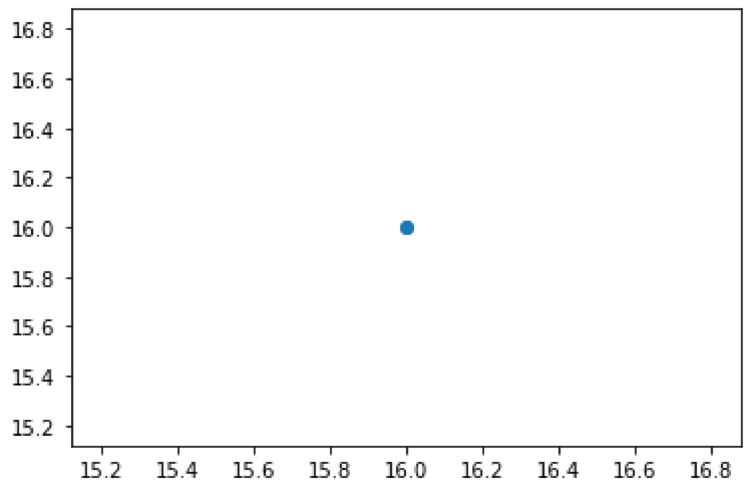
In [24]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[24]:

	Co-efficient
Sr. No.	0.0

In [25]:
```
prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[25]:
```
<matplotlib.collections.PathCollection at 0x2a130271280>
```



In [26]:
```
print(lr.score(x_test,y_test))
```

```
1.0
```

RIDGE AND LASSO REGRESSION

In [27]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [28]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[28]:
```
Ridge(alpha=10)
```

In [29]:
```
rr.score(x_test,y_test)
```

Out[29]:
```
1.0
```

In [30]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_des
cent.py:530: ConvergenceWarning: Objective did not converge. You might want to
increase the number of iterations. Duality gap: 0.0, tolerance: 0.0
  model = cd_fast.enet_coordinate_descent(
```

Out[30]:
```
Lasso(alpha=10)
```

In [31]:
```
la.score(x_test,y_test)
```

Out[31]:
```
1.0
```

In [32]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_des
cent.py:530: ConvergenceWarning: Objective did not converge. You might want to
increase the number of iterations. Duality gap: 0.0, tolerance: 0.0
  model = cd_fast.enet_coordinate_descent(
```

Out[32]:
```
ElasticNet()
```

In [33]:
```
print(en.coef_)
```

```
[0.]
```

In [34]:
```
print(en.predict(x_test))
```

```
[16. 16.]
```

In [35]:
```
print(en.score(x_test,y_test))
```

```
1.0
```

EVALUATION METRICS

In [36]:
```
from sklearn import metrics
```

In [37]:
```
print("Mean Absolute error",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute error 0.0
```

In [38]:
```
print("Mean Squared error",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared error 0.0
```

In [39]:
```
print("Root Mean Absolute error",np.sqrt(metrics.mean_squared_error(y_test,predic
```

```
Root Mean Absolute error 0.0
```

MODEL SAVING ¶

In [40]:
```
import pickle
```

In [42]:
```
filename='predict4'
pickle.dump(lr,open(filename,'wb'))
```