Data Cleaning

In [1]:

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

```
In [2]: df = pd.read_csv(r"C:\Users\user\Downloads\23_Vande Bharat.csv")
        df
```

Out[2]:

ing ion	Terminal City	Terminal Station	Operator	No. of Cars	Frequency	Distance	Travel
elhi	Varanasi	Varanasi Junction	NR	16	Except Thursdays	759 km (472 mi)	08l
elhi	Katra	Shri Mata Vaishno Devi Katra	NR	16	Except Tuesdays	655 km (407 mi)	08l
ıbai ıtral	Gandhinagar	Gandhinagar Capital	WR	16	Except Wednesdays	522 km (324 mi)	06l
elhi	Andaura	Amb Andaura	NR	16	Except Fridays	412 km (256 mi)	05l
ınai ıtral	Mysuru	Mysore Junction	SR	16	Except Wednesdays	496 km (308 mi)	06l
pur tion	Nagpur	Nagpur Junction	SECR	8	Except Saturdays	412 km (256 mi)	05l
rah tion	Siliguri	New Jalpaiguri Junction	ER	16	Except Wednesdays	565 km (351 mi)	07l
ıam tion	Hyderabad	Secunderabad Junction	ECoR	16	Except Sundays	698 km (434 mi)	08l
pati vaji nus	Solapur	Solapur	CR	16	Except Wednesdays (22225) , Except Thursdays (...	452 km (281 mi)	06l
pati vaji nus	Shirdi	Sainagar Shirdi	CR	16	Except Tuesdays	339 km (211 mi)	05l
janj Rani pati)	Delhi	Hazrat Nizamuddin	WCR	16	Except Saturdays	702 km (436 mi)	07l
bad tion	Tirupati	Tirupati	SCR	16	Except Tuesdays	661 km (411 mi)	08l
ınai ıtral	Coimbatore	Coimbatore Junction	SR	8	Except Wednesdays	495 km (308 mi)	05l
elhi ıent	Ajmer	Ajmer Junction	NWR	16	Except Wednesdays	428 km (266 mi)	05l
god	Thiruvananthapuram	Thiruvananthapuram Central	SR	16	Except Thursdays	587 km (365 mi)	08l
rah tion	Puri	Puri	SER	16	Except Thursdays	500 km (310 mi)	06l

ing ion	Terminal City	Terminal Station	Operator	No. of Cars	Frequency	Distance	Travel
har inal	Dehradun	Dehradun Terminal	NR	8	Except Wednesdays	304 km (189 mi)	04h
guri tion	Guwahati	Guwahati	NFR	8	Except Tuesdays	407 km (253 mi)	05h
pati vaji nus	Madgaon	Madgaon Junction	CR	16	Except Fridays\n(Non-Monsoon)	586 km (364 mi)	07h 45m\n Mon
pati vaji nus	Madgaon	Madgaon Junction	CR	16	Monday, Wednesday, Friday (22229)\nTuesday, Th...	586 km (364 mi)	05m\n(Mon
tion	Ranchi	Ranchi Junction	ECR	8	Except Tuesdays	379 km (235 mi)	06h
City	Hubbali - Dharwad	Dharwad	SWR	8	Except Tuesdays	490 km (300 mi)	06h
ganj Rani pati)	Jabalpur	Jabalpur Junction	WCR	8	Except Tuesdays	337 km (209 mi)	04h
tion	Bhopal	Bhopal Junction	WR	8	Except Sundays	250 km (160 mi)	03h
pur tion	Ahmedabad	Sabarmati Junction	NWR	8	Except Tuesdays	449 km (279 mi)	06h
pur tion	Charbagh	Lucknow Charbagh	NER	8	Except Saturdays	296 km (184 mi)	04h

In [3]:

```
# to display info
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 [4]:

```
# t display summerize the data
df.describe()
```

Out[4]:

	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 [5]:

```
# to display columes
df.columns
```

Out[5]:

```
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')
```

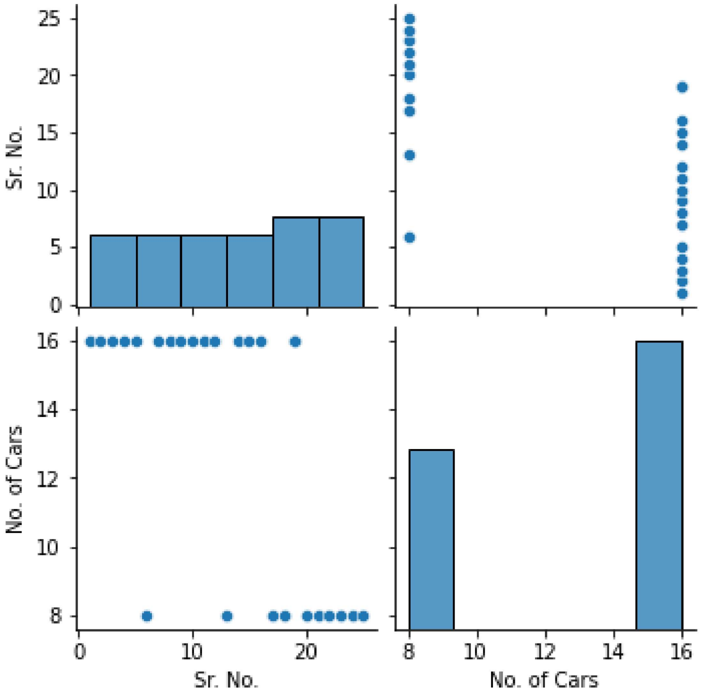
In [6]:
```
df.isna().sum()
```

Out[6]:
```
Sr. No.                0
Train Name             0
Train Number           0
Originating City       0
Originating Station    0
Terminal City          0
Terminal Station       0
Operator               0
No. of Cars            0
Frequency              0
Distance               0
Travel Time            0
Speed                  0
Average Speed          0
Inauguration           0
Average occupancy      0
dtype: int64
```

EDA and visualization

In [7]:
```
sns.pairplot(df)
```

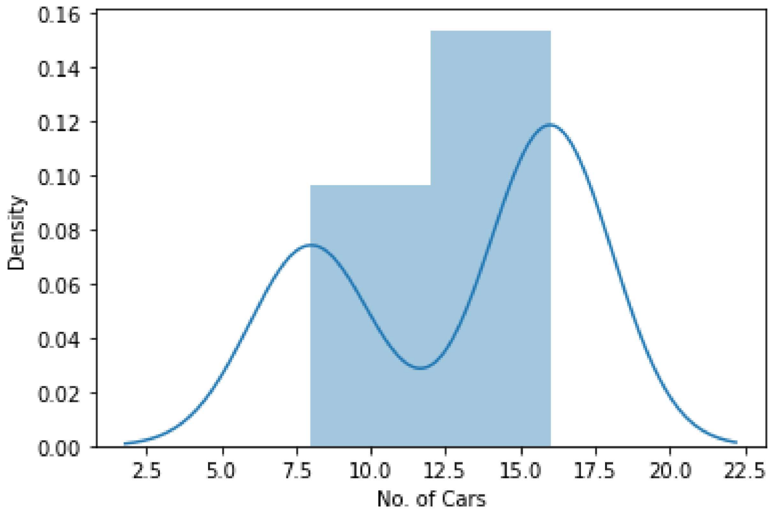
Out[7]:
```
<seaborn.axisgrid.PairGrid at 0x1e901d2e700>
```



```
In [8]: # to display distribution graph for price column
        sns.distplot(df['No. of Cars'])
```

```
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)
```

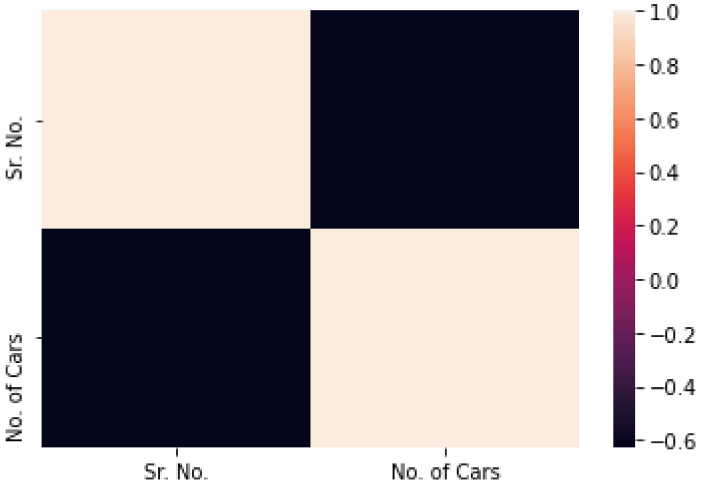
```
Out[8]: <AxesSubplot:xlabel='No. of Cars', ylabel='Density'>
```



```
In [10]: df1 = df[['Sr. No.','No. of Cars']]
```

```
In [11]: # correlation map to find relationship
         sns.heatmap(df1.corr())
```

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



In [12]:

```
# Assign x and y for linear regression
x = df1[['Sr. No.']]
y = df1['No. of Cars']
```

In [13]:

```
# to split dataset into training data and test data

from sklearn.model_selection import train_test_split

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

In [14]:

```
#Linear Regression

from sklearn.linear_model import LinearRegression

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

Out[14]:

```
LinearRegression()
```

In [15]:

```
# intercept is value of c
print(lr.intercept_)
```

```
16.88360062079669
```

In [16]:

```
# co-efficient value of m
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[16]:

	Co-efficient
Sr. No.	-0.302121

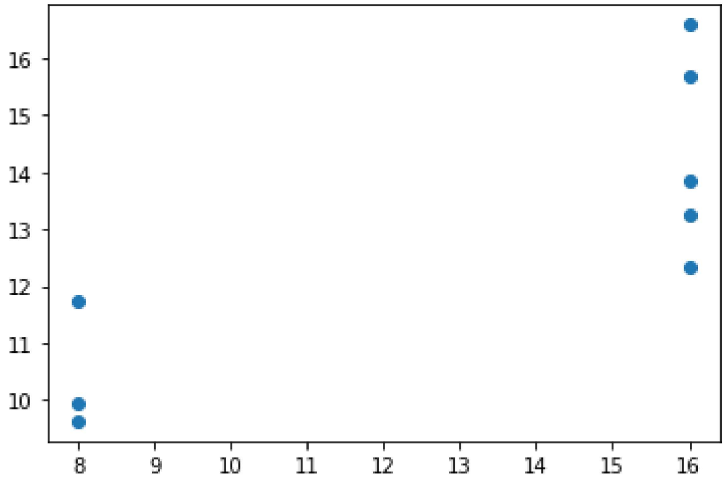
In [17]:

```
#predict the graph in linear regression graph

prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[17]:

```
<matplotlib.collections.PathCollection at 0x1e9042f8430>
```



```
In [18]: #Accuracy of linear regression

         print(lr.score(x_test,y_test))

         0.6142203728330706
```

```
In [19]: lr.score(x_train,y_train)

Out[19]: 0.286426455076222
```

```
In [20]: from sklearn.linear_model import Ridge,Lasso
```

```
In [21]: rr = Ridge(alpha=10)
         rr.fit(x_train,y_train)
         rr.score(x_test,y_test)

Out[21]: 0.6112574064680383
```

```
In [22]: rr.score(x_train,y_train)

Out[22]: 0.2863885355830754
```

```
In [23]: lr = Lasso(alpha=10)
         lr.fit(x_train,y_train)
         lr.score(x_test,y_test)

Out[23]: 0.2645730167188147
```

```
In [24]: lr.score(x_train,y_train)

Out[24]: 0.14867459235022595
```

Elastic

```
In [25]: from sklearn.linear_model import ElasticNet
         es = ElasticNet()
         es.fit(x_train,y_train)

Out[25]: ElasticNet()
```

```
In [26]: print(es.coef_)

         [-0.28862153]
```

```
In [27]: print(es.intercept_)

         16.70510687316012
```

```
In [28]: print(es.score(x_test,y_test))

0.6021777965397894
```

Evaluation Model

```
In [29]: from sklearn import metrics
```

```
In [30]: print("Mean absolute Error:",metrics.mean_absolute_error(y_test,prediction))

Mean absolute Error: 2.093636833936885
```

```
In [31]: print("Mean squared Error:",metrics.mean_squared_error(y_test,prediction))

Mean squared Error: 5.7866944075039415
```

```
In [32]: print("Root Mean squared Error:",np.sqrt(metrics.mean_squared_error(y_test,prec

Root Mean squared Error: 2.4055549063581863
```

model saving

```
In [33]: import pickle # pickle is used to model saving
```

```
In [34]: filename ="23_Vande Bharat prediction"
         pickle.dump(lr,open(filename,'wb'))
```

```
In [ ]:
```