1. Import Libraries

In [4]:

```
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

2. Import DataSet

In [5]:

```
data=pd.read_csv("Newspaper.csv")
data.head()
```

Out[5]:

	Newspaper	daily	sunday
0	Baltimore Sun	391.952	488.506
1	Boston Globe	516.981	798.298
2	Boston Herald	355.628	235.084
3	Charlotte Observer	238.555	299.451
4	Chicago Sun Times	537.780	559.093

3. Data Undestadning

In [6]:

```
data=data.drop(['Newspaper'],axis=1)
```

In [6]:

```
data.isnull().sum()
```

Out[6]:

```
daily     0
sunday    0
dtype: int64
```

In [7]:

```
data.dtypes
```

Out[7]:

```
daily     float64
sunday    float64
dtype: object
```

4. Data Preparing

1. Assumption Check

In [9]:

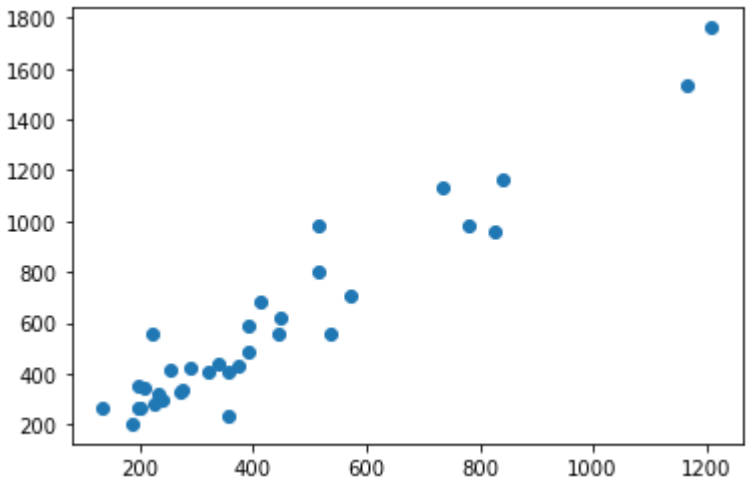
```
import matplotlib.pyplot as plt
```

In [13]:

```
plt.scatter(x=data['daily'],y=data['sunday'],)
```

Out[13]:

```
<matplotlib.collections.PathCollection at 0x156bb8da190>
```



In [14]:

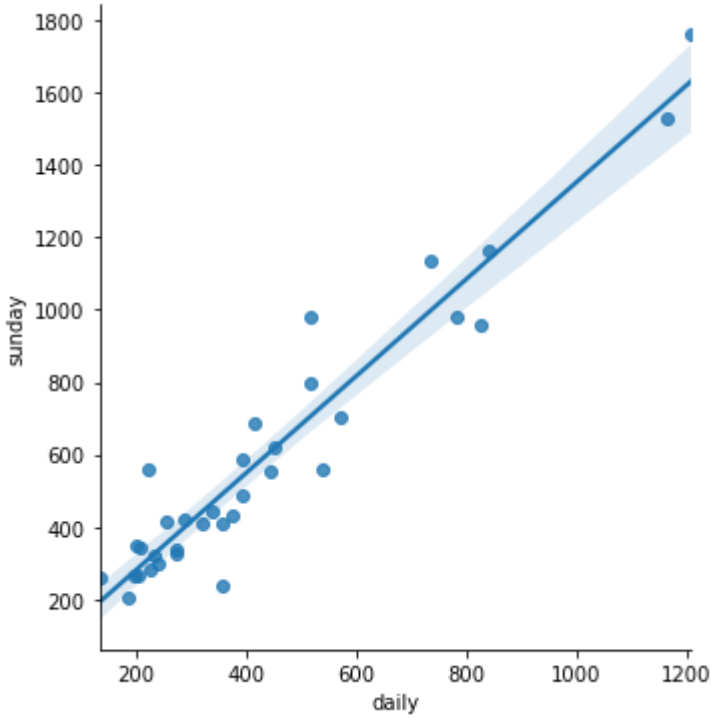
```
import seaborn as sns
```

In [17]:

```
sns.lmplot(x='daily',y='sunday',data=data)
```

Out[17]:

```
<seaborn.axisgrid.FacetGrid at 0x156bb6ab070>
```



2. Check Data is Linearlly Distributed Or not

In [19]:

```
data.skew()
```

Out[19]:

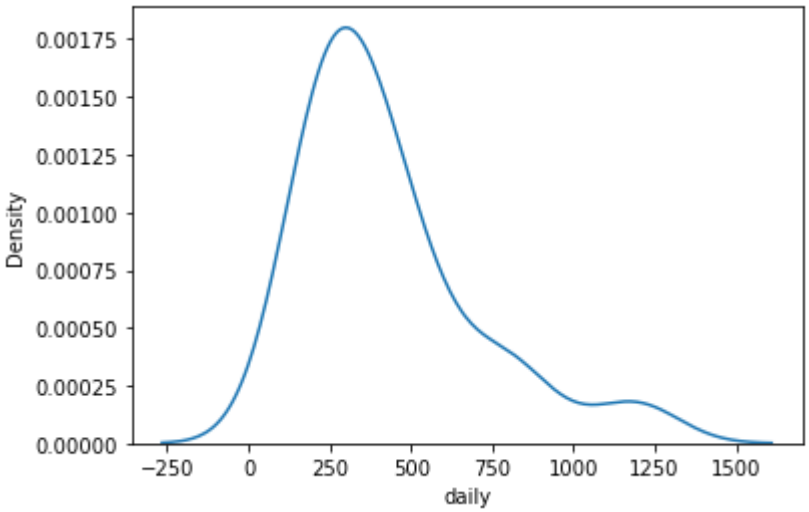
```
daily     1.532159
sunday    1.564473
dtype: float64
```

In [24]:

```
sns.distplot(a=data['daily'],hist=False)
```

Out[24]:

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



3 Homoscadacity | 4. Zero Mean Residual Test

In [25]:

```
# this test will be done after build the model
```

5. Auto Regression

In [7]:

```
#no auto regression
```

6. Multi Colinearity

In [1]:

```
# no multi co linearity due to less input feature
```

7. Model Bulding

In [2]:

```
# stats model
#sklearn
```

7.1 Using State Model For Linear Regression

In [3]:

```
import statsmodels.formula.api as sfa
```

In [12]:

```
l_model=sfa.ols(formula='sunday~daily',data=data).fit()
```

In [13]:

```
l_model.params
```

Out[13]:

```
Intercept    13.835630
daily         1.339715
dtype: float64
```

7. Model Testing

In [14]:

```
x=200
```

In [16]:

```
b=200*1.33+13.835
b
```

Out[16]:

```
279.835
```

8. Machine Prediction

In [29]:

```
test_data=pd.DataFrame({'daily':[200,300,400]})
```

9. Model Deployment

In [30]:

```
from pickle import dump
```

In [33]:

```
dump(l_model,open('model.apk','wb'))
```

In [31]:

```
from pickle import load
```

In [38]:

```
loaded=load(open("model.apk",'rb'))
```

In [41]:

```
y_pred=loaded.predict(test_data)
y_pred
```

Out[41]:

```
0    281.778581
1    415.750057
2    549.721533
dtype: float64
```