LINEAR REGRESSION

In [ ]:

```
```

In [3]:

```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [4]:

```
newspaper_data=pd.read_csv('NewspaperData (1).csv')
newspaper_data
```

Out[4]:

	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
5	Chicago Tribune	733.775	1133.249
6	Cincinnati Enquirer	198.832	348.744
7	Denver Post	252.624	417.779
8	Des Moines Register	206.204	344.522
9	Hartford Courant	231.177	323.084
10	Houston Chronicle	449.755	620.752
11	Kansas City Star	288.571	423.305
12	Los Angeles Daily News	185.736	202.614
13	Los Angeles Times	1164.388	1531.527
14	Miami Herald	444.581	553.479
15	Minneapolis Star Tribune	412.871	685.975
16	New Orleans Times-Picayune	272.280	324.241
17	New York Daily News	781.796	983.240
18	New York Times	1209.225	1762.015
19	Newsday	825.512	960.308
20	Omaha World Herald	223.748	284.611
21	Orange County Register	354.843	407.760
22	Philadelphia Inquirer	515.523	982.663
23	Pittsburgh Press	220.465	557.000
24	Portland Oregonian	337.672	440.923
25	Providence Journal-Bulletin	197.120	268.060
26	Rochester Democrat & Chronicle	133.239	262.048
27	Rocky Mountain News	374.009	432.502
28	Sacramento Bee	273.844	338.355
29	San Francisco Chronicle	570.364	704.322
30	St. Louis Post-Dispatch	391.286	585.681
31	St. Paul Pioneer Press	201.860	267.781
32	Tampa Tribune	321.626	408.343
33	Washington Post	838.902	1165.567

Data understanding

3.1 perform initial analysis

In [5]:

```
newspaper_data.shape
```

Out[5]:

```
(34, 3)
```

In [10]:

```
newspaper_data.describe()
```

Out[10]:

	daily	sunday
count	34.000000	34.000000
mean	430.962471	591.202412
std	269.211470	376.418051
min	133.239000	202.614000
25%	233.021500	327.769500
50%	355.235500	436.712500
75%	516.616500	699.735250
max	1209.225000	1762.015000

In [10]:

```
newspaper_data.isna().sum()
```

Out[10]:

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

3.2 Assumption check

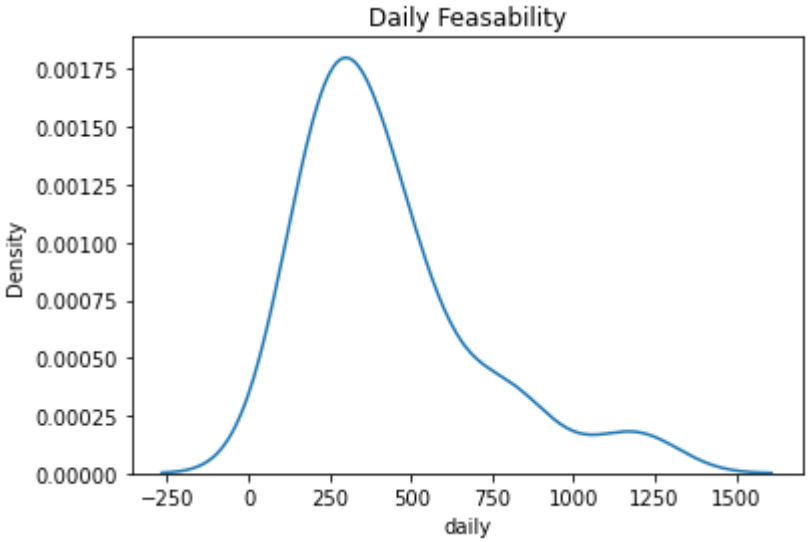
1. Normality test

In [12]:

```
sns.distplot(a=newspaper_data['daily'], hist=False)
plt.title('Daily Feasability')
```

Out[12]:

```
Text(0.5, 1.0, 'Daily Feasability')
```



In [14]:

```
newspaper_data['daily'].skew() #checking skewness
```

Out[14]:

```
1.5321591323040094
```

In [15]:

```
newspaper_data['daily'].kurtosis() #measure of outliers
```

Out[15]:

```
1.999034084097406
```

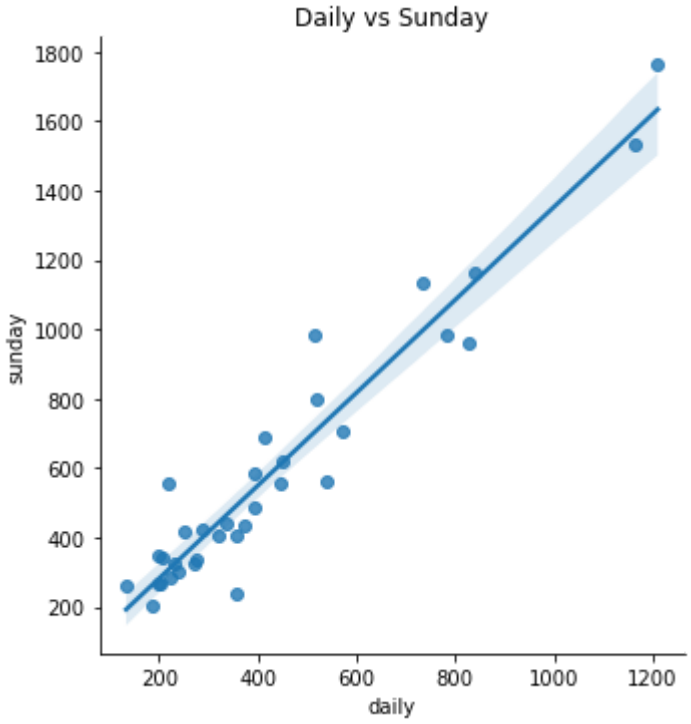
LINEARITY TEST

In [16]:

```
sns.lmplot(x='daily',y='sunday',data=newspaper_data)
plt.title('Daily vs Sunday')
plt.show
```

Out[16]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```

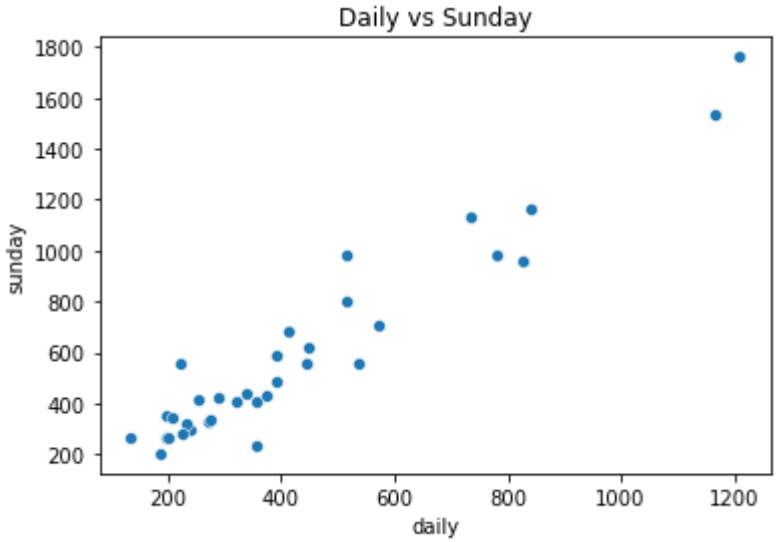


In [17]:

```
sns.scatterplot(x='daily',y='sunday',data=newspaper_data)
plt.title('Daily vs Sunday')
plt.show
```

Out[17]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



4.Data preparation

In [19]:

```
del newspaper_data['Newspaper']
```

In [21]:

```
newspaper_data
```

Out[21]:

	daily	sunday
0	391.952	488.506
1	516.981	798.298
2	355.628	235.084
3	238.555	299.451
4	537.780	559.093
5	733.775	1133.249
6	198.832	348.744
7	252.624	417.779
8	206.204	344.522
9	231.177	323.084
10	449.755	620.752
11	288.571	423.305
12	185.736	202.614
13	1164.388	1531.527
14	444.581	553.479
15	412.871	685.975
16	272.280	324.241
17	781.796	983.240
18	1209.225	1762.015
19	825.512	960.308
20	223.748	284.611
21	354.843	407.760
22	515.523	982.663
23	220.465	557.000
24	337.672	440.923
25	197.120	268.060
26	133.239	262.048
27	374.009	432.502
28	273.844	338.355
29	570.364	704.322
30	391.286	585.681
31	201.860	267.781
32	321.626	408.343
33	838.902	1165.567

In [24]:

```
newspaper_data.isna().sum()
```

Out[24]:

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

In [25]:

```
newspaper_data.dtypes
```

Out[25]:

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

5.Model Building.

it can be performed by using any of the following 2 libraries. 1.Statsmodels-ols 2.Sklearn-LinearRegression

5.1 Using stats models build Linear Regression

In [26]:

```
import statsmodels.formula.api as smf
```

In [32]:

```
linear_model=smf.ols(formula='sunday~daily',data=newspaper_data)
linear_model
```

Out[32]:

```
<statsmodels.regression.linear_model.OLS at 0x1b8cba15df0>
```

6.Model Training

In [35]:

```
linear_model=linear_model.fit()
```

In [37]:

```
linear_model.params
```

Out[37]:

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

7. Model Testing

Manual prediction

In [51]:

```
#if daily sales=300,sunday sales?--->415.75012999999996
#(1.339715*300)+13.835630
#if daily sales=250,sunday sales?----->348.76437999999996
#(1.339715*250)+13.835630
#if daily sales=585,sunday sales?------>797.5689050000001
#(1.339715*585)+13.835630
```

Out[51]:

```
797.5689050000001
```

Automatic prediction

In [53]:

```
x_test=pd.DataFrame(data={'daily':[300,250,585]})
x_test
```

Out[53]:

	daily
0	300
1	250
2	585

In [54]:

```
linear_model.predict(x_test)
```

Out[54]:

```
0    415.750057
1    348.764319
2    797.568763
dtype: float64
```

8. Model Evaluation

In [ ]:

In [ ]:

In [ ]:

```
```

9.Model Deployement

In [66]:

```
from pickle import dump
```

In [67]:

```
dump(linear_model,open('linear_intelligence.pkl','wb'))
```

In [68]:

```
from pickle import load
```

In [69]:

```
loaded_lin_model=load(open('linear_intelligence.pkl','rb'))
```

In [70]:

```
loaded_lin_model.predict(x_test)
```

Out[70]:

```
0    415.750057
1    348.764319
2    797.568763
dtype: float64
```

In [ ]:

```
```

In [ ]:

```
```