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

```
In [2]: path1 ='.\Marketing Data[1].csv'
        path2 ='.\Revenue Data 1[1].csv'
        path3 ='.\Revenue Data 2[1].csv'
        path4 ='.\Visitors Data[1].xlsx'
```

```
In [ ]:
```

```
In [3]: df1=pd.read_csv(path1)
        df2=pd.read_csv(path2)
        df3=pd.read_csv(path3)
        df4=pd.read_excel(path4)
```

```
In [4]: df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 185 entries, 0 to 184
Data columns (total 3 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Date             185 non-null    object
 1   Marketing Spend  178 non-null    float64
 2   Promo            182 non-null    object
dtypes: float64(1), object(2)
memory usage: 4.5+ KB
```

```
In [5]: df1
```

Out[5]:

	Date	Marketing Spend	Promo
0	09/11/2020	651.375	No Promo
1	10/11/2020	1298.250	Promotion Red
2	11/11/2020	1559.375	Promotion Blue
3	12/11/2020	1801.750	No Promo
4	13/11/2020	2614.500	No Promo
...	...	...	...
180	08/05/2021	1121.875	No Promo
181	09/05/2021	871.000	No Promo
182	10/05/2021	NaN	NaN
183	11/05/2021	NaN	NaN
184	12/05/2021	NaN	NaN

185 rows × 3 columns

```
In [6]: df1['Date']=pd.to_datetime(df1['Date'],format='%d/%m/%Y')
```

```
In [7]: df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 185 entries, 0 to 184
Data columns (total 3 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Date             185 non-null    datetime64[ns]
 1   Marketing Spend  178 non-null    float64
 2   Promo            182 non-null    object
dtypes: datetime64[ns](1), float64(1), object(1)
memory usage: 4.5+ KB
```

```
In [8]: df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 133 entries, 0 to 132
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Date          133 non-null    object
 1   Week_ID       129 non-null    float64
 2   Month Number  129 non-null    float64
 3   Month_ID      129 non-null    float64
 4   Year          133 non-null    int64
 5   Day_Name      129 non-null    object
 6   Revenue       126 non-null    float64
dtypes: float64(4), int64(1), object(2)
memory usage: 7.4+ KB
```

```
In [9]: df2['Date']=pd.to_datetime(df2['Date'],format='%d/%m/%Y')
```

```
In [10]: df3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 56 entries, 0 to 55
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Date          56 non-null     object
 1   Week_ID       53 non-null     float64
 2   Month Number  53 non-null     float64
 3   Month_ID      53 non-null     float64
 4   Year          53 non-null     float64
 5   Day_Name      53 non-null     object
 6   Revenue       52 non-null     float64
dtypes: float64(5), object(2)
memory usage: 3.2+ KB
```

```
In [11]: df3['Date']=pd.to_datetime(df3['Date'],format='%d/%m/%Y')
```

```
In [12]: df4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 185 entries, 0 to 184
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Date      185 non-null    datetime64[ns]
 1   Visitors  177 non-null    float64
dtypes: datetime64[ns](1), float64(1)
memory usage: 3.0 KB
```

```
In [ ]:
```

```
In [13]: df_rev=pd.concat([df2,df3])
         df_rev
```

Out[13]:

	Date	Week_ID	Month Number	Month_ID	Year	Day_Name	Revenue
0	2020-11-09	34.0	11.0	11.0	2020.0	Monday	465.0
1	2020-11-10	34.0	11.0	11.0	2020.0	Tuesday	10386.0
2	2020-11-11	34.0	11.0	11.0	2020.0	Wednesday	12475.0
3	2020-11-12	34.0	11.0	11.0	2020.0	Thursday	11712.0
4	2020-11-13	34.0	11.0	11.0	2020.0	Friday	10000.0
...	...	...	...	...	...	...	...
51	2021-05-08	60.0	5.0	17.0	2021.0	Saturday	5927.0
52	2021-05-09	61.0	5.0	17.0	2021.0	Sunday	3861.0
53	2021-05-10	NaN	NaN	NaN	NaN	NaN	NaN
54	2021-05-11	NaN	NaN	NaN	NaN	NaN	NaN
55	2021-05-12	NaN	NaN	NaN	NaN	NaN	NaN

189 rows × 7 columns

```
In [14]: df_m_r = df1.merge(df_rev,on='Date',how='left')
         df_m_r
```

Out[14]:

	Date	Marketing Spend	Promo	Week_ID	Month Number	Month_ID	Year	Day_Name	Revenue
0	2020-11-09	651.375	No Promo	34.0	11.0	11.0	2020.0	Monday	465.0
1	2020-11-10	1298.250	Promotion Red	34.0	11.0	11.0	2020.0	Tuesday	10386.0
2	2020-11-11	1559.375	Promotion Blue	34.0	11.0	11.0	2020.0	Wednesday	12475.0
3	2020-11-12	1801.750	No Promo	34.0	11.0	11.0	2020.0	Thursday	11712.0
4	2020-11-13	2614.500	No Promo	34.0	11.0	11.0	2020.0	Friday	10000.0
...	...	...	...	...	...	...	...	...	...
184	2021-05-08	1121.875	No Promo	60.0	5.0	17.0	2021.0	Saturday	5927.0
185	2021-05-09	871.000	No Promo	61.0	5.0	17.0	2021.0	Sunday	3861.0
186	2021-05-10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
187	2021-05-11	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
188	2021-05-12	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

189 rows × 9 columns

```
In [15]: df_m_r
```

Out[15]:

	Date	Marketing Spend	Promo	Week_ID	Month Number	Month_ID	Year	Day_Name	Revenue
0	2020-11-09	651.375	No Promo	34.0	11.0	11.0	2020.0	Monday	465.0
1	2020-11-10	1298.250	Promotion Red	34.0	11.0	11.0	2020.0	Tuesday	10386.0
2	2020-11-11	1559.375	Promotion Blue	34.0	11.0	11.0	2020.0	Wednesday	12475.0
3	2020-11-12	1801.750	No Promo	34.0	11.0	11.0	2020.0	Thursday	11712.0
4	2020-11-13	2614.500	No Promo	34.0	11.0	11.0	2020.0	Friday	10000.0
...	...	...	...	...	...	...	...	...	...
184	2021-05-08	1121.875	No Promo	60.0	5.0	17.0	2021.0	Saturday	5927.0
185	2021-05-09	871.000	No Promo	61.0	5.0	17.0	2021.0	Sunday	3861.0
186	2021-05-10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
187	2021-05-11	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
188	2021-05-12	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

189 rows × 9 columns

```
In [16]: df = df_m_r.merge(df4,on='Date',how='left')
         df
```

Out[16]:

	Date	Marketing Spend	Promo	Week_ID	Month Number	Month_ID	Year	Day_Name	Revenue	Visitors
0	2020-11-09	651.375	No Promo	34.0	11.0	11.0	2020.0	Monday	465.0	707.0
1	2020-11-10	1298.250	Promotion Red	34.0	11.0	11.0	2020.0	Tuesday	10386.0	1455.0
2	2020-11-11	1559.375	Promotion Blue	34.0	11.0	11.0	2020.0	Wednesday	12475.0	1520.0
3	2020-11-12	1801.750	No Promo	34.0	11.0	11.0	2020.0	Thursday	11712.0	1726.0
4	2020-11-13	2614.500	No Promo	34.0	11.0	11.0	2020.0	Friday	10000.0	2134.0
...	...	...	...	...	...	...	...	...	...	...
184	2021-05-08	1121.875	No Promo	60.0	5.0	17.0	2021.0	Saturday	5927.0	1483.0
185	2021-05-09	871.000	No Promo	61.0	5.0	17.0	2021.0	Sunday	3861.0	1303.0
186	2021-05-10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
187	2021-05-11	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
188	2021-05-12	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

189 rows × 10 columns

```
In [17]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 189 entries, 0 to 188
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Date             189 non-null    datetime64[ns]
 1   Marketing Spend  182 non-null    float64
 2   Promo            186 non-null    object
 3   Week_ID          182 non-null    float64
 4   Month Number     182 non-null    float64
 5   Month_ID         182 non-null    float64
 6   Year             186 non-null    float64
 7   Day_Name         182 non-null    object
 8   Revenue          178 non-null    float64
 9   Visitors         181 non-null    float64
dtypes: datetime64[ns](1), float64(7), object(2)
memory usage: 16.2+ KB
```

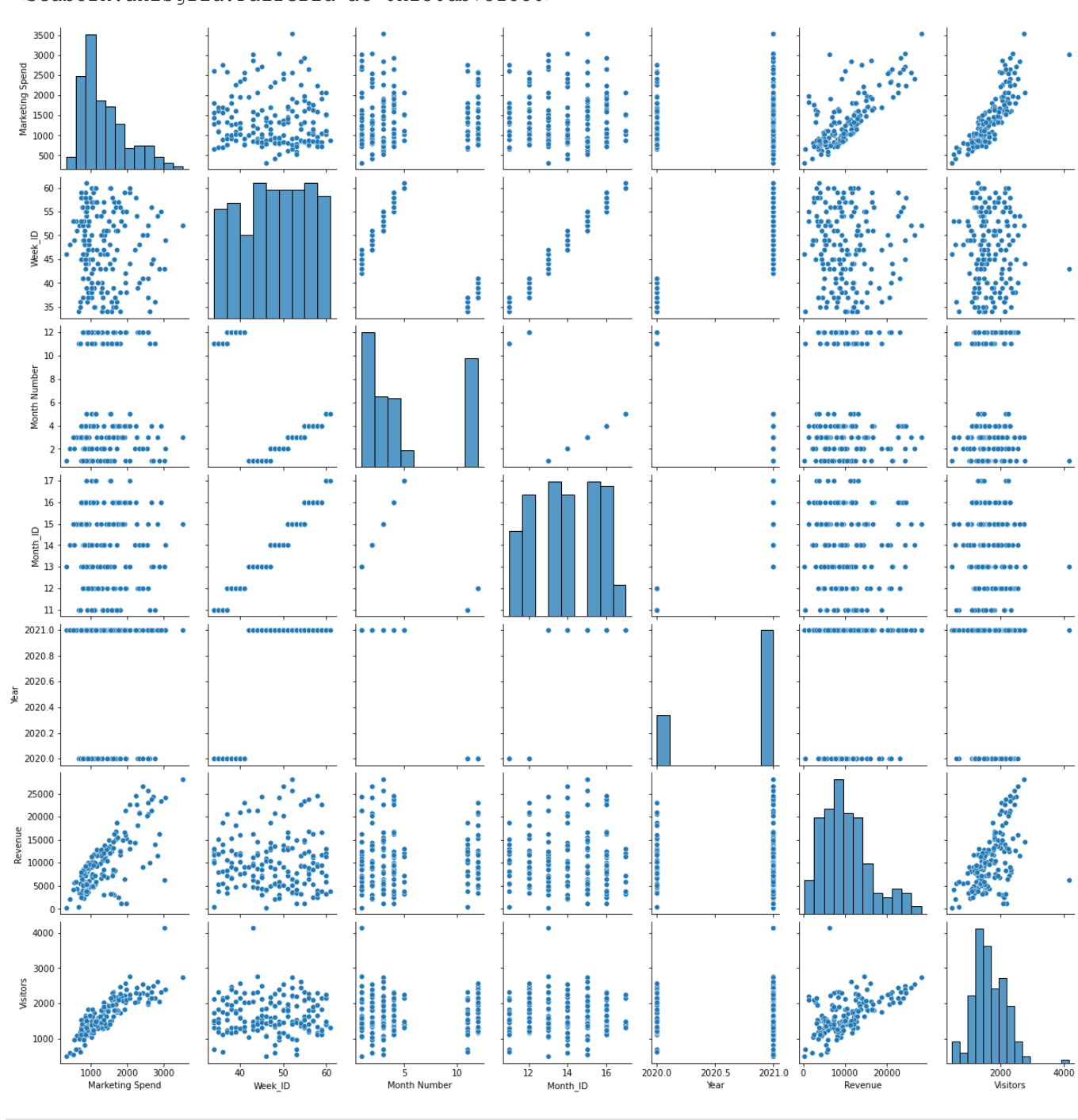
```
In [18]: df.dropna(inplace=True)
```

```
In [19]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 169 entries, 0 to 185
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Date             169 non-null    datetime64[ns]
 1   Marketing Spend  169 non-null    float64
 2   Promo            169 non-null    object
 3   Week_ID          169 non-null    float64
 4   Month Number     169 non-null    float64
 5   Month_ID         169 non-null    float64
 6   Year             169 non-null    float64
 7   Day_Name         169 non-null    object
 8   Revenue          169 non-null    float64
 9   Visitors         169 non-null    float64
dtypes: datetime64[ns](1), float64(7), object(2)
memory usage: 14.5+ KB
```

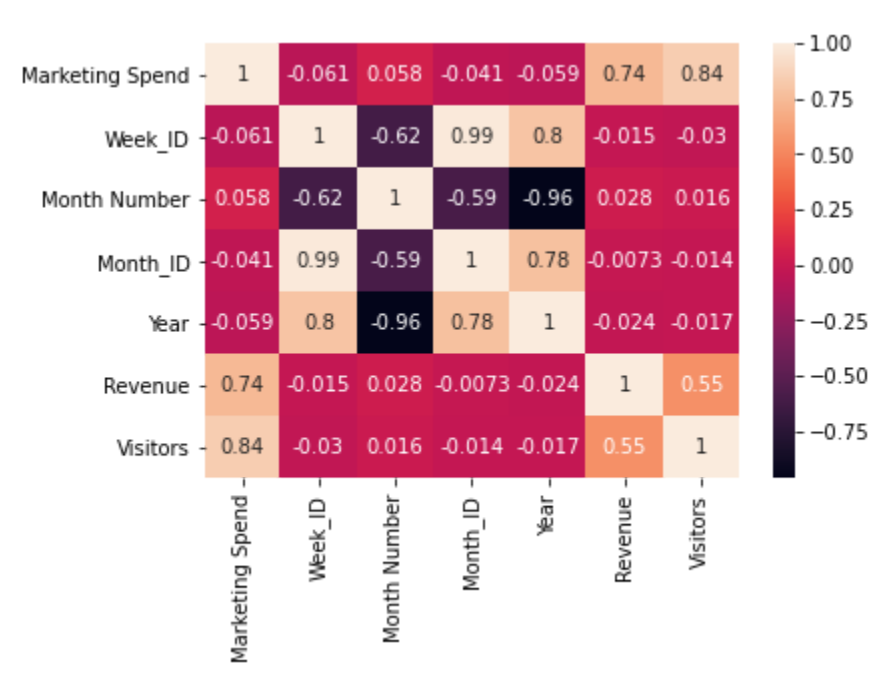
```
In [20]: sns.pairplot(df)
```

Out[20]: <seaborn.axisgrid.PairGrid at 0x230db73f550>




```
In [21]: sns.heatmap(df.corr(),annot = True)
```

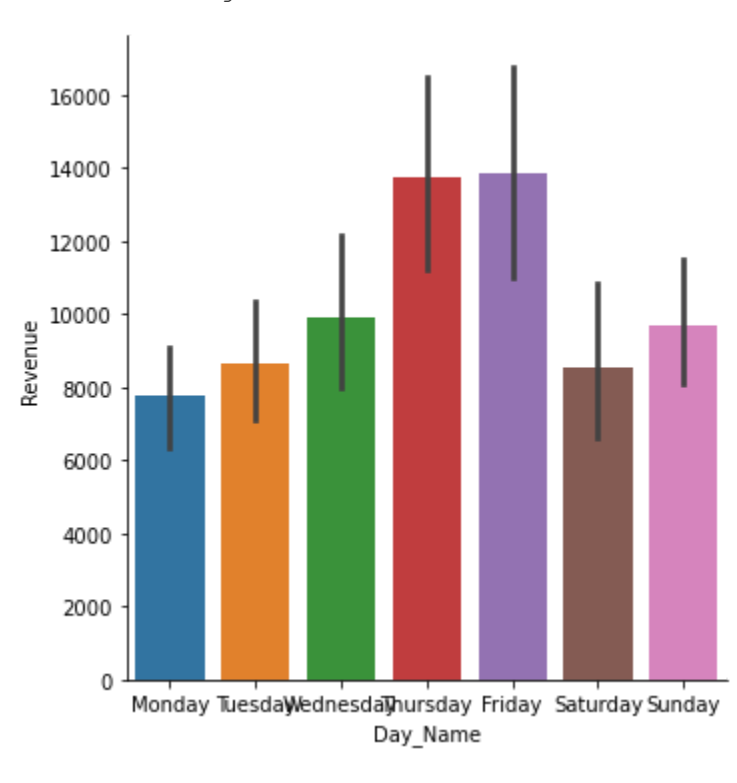
Out[21]: <AxesSubplot:>



Revenue and Visitors have positive correlation with Marketing Spend,

```
In [22]: sns.catplot(x='Day_Name',y='Revenue',data = df,kind = 'bar')
```

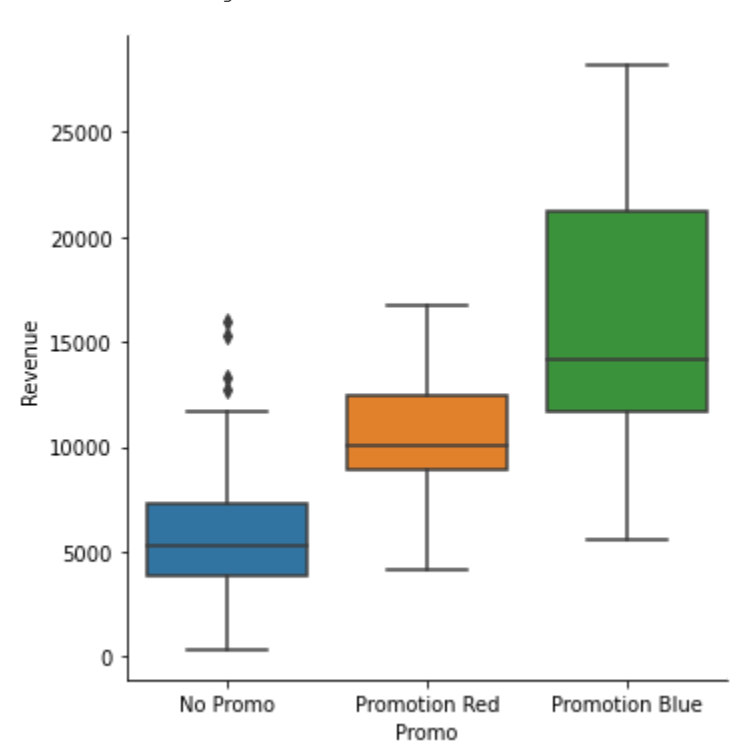
Out[22]: <seaborn.axisgrid.FacetGrid at 0x230dce70e50>



Thursday and Friday are the most revenue days

```
In [23]: sns.catplot(x='Promo',y='Revenue',data = df,kind = 'box')
```

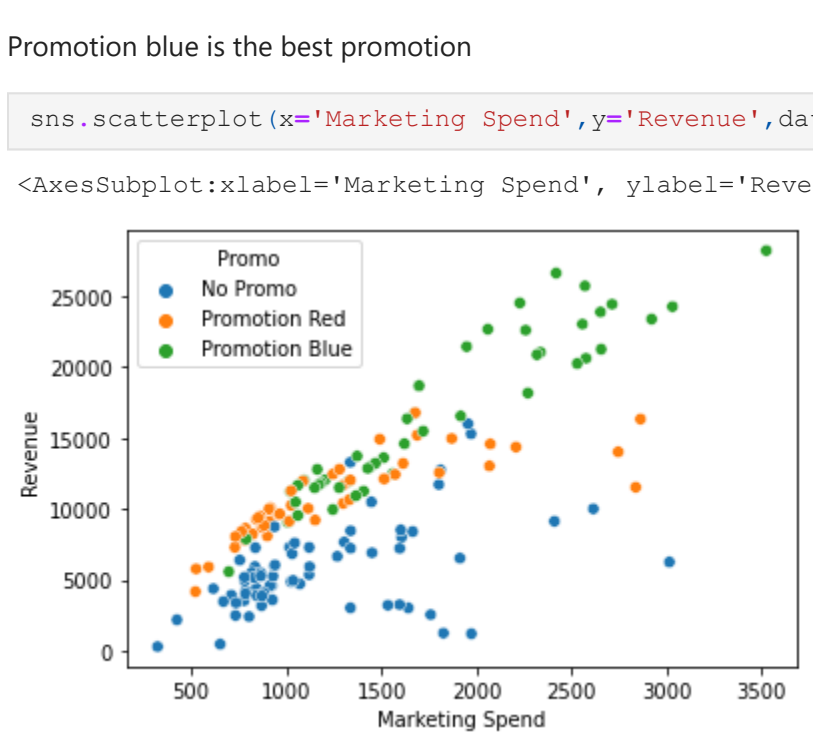
Out[23]: <seaborn.axisgrid.FacetGrid at 0x230de679220>



Promotion blue is the best promotion

```
In [25]: sns.scatterplot(x='Marketing Spend',y='Revenue',data = df,hue ='Promo')
```

Out[25]: <AxesSubplot:xlabel='Marketing Spend', ylabel='Revenue'>



Revenue is directly proportional to Marketing Spend even for higher values of marketing spend in case of Promotion blue. whereas revenue for Promotion red starts to flatten after Marketing spend of £2000.

```
In [26]: df.columns
```

```
Out[26]: Index(['Date', 'Marketing Spend', 'Promo', 'Week_ID', 'Month Number',
                'Month_ID', 'Year', 'Day_Name', 'Revenue', 'Visitors'],
               dtype='object')
```

```
In [27]: features=['Marketing Spend','Visitors', 'Day_Name','Promo']
         X=df[features]
         y=df['Revenue']
```

```
In [28]: from sklearn.model_selection import train_test_split
         X_train, X_valid, y_train, y_valid = train_test_split(X, y,train_size=0.8,test_size=0
```

```
In [29]: from sklearn.preprocessing import OrdinalEncoder

         ordinal_encoder = OrdinalEncoder()
         label_X_train = X_train.copy()
         label_X_valid = X_valid.copy()

         label_X_train[['Promo','Day_Name']] = ordinal_encoder.fit_transform(X_train[['Promo',
         label_X_valid[['Promo','Day_Name']] = ordinal_encoder.transform(X_valid[['Promo','Day_
```

```
In [30]: from sklearn.linear_model import LinearRegression
         from sklearn.metrics import mean_absolute_error
         def mae_lm(X_train, X_valid, y_train, y_valid):
             model = LinearRegression()
             model.fit(X_train, y_train)
             preds = model.predict(X_valid)
             return mean_absolute_error(y_valid, preds)
```

```
In [31]: print('Mean absolute error for Linear Regression model is {}'.format(mae_lm(label_X_tr
```

```
Mean absolute error for Linear Regression model is 2519.85161049745
```

```
In [32]: from sklearn.ensemble import RandomForestRegressor
         def mae_rf(X_train, X_valid, y_train, y_valid):
             model = RandomForestRegressor(n_estimators=200,random_state=0)
             model.fit(X_train, y_train)
             preds = model.predict(X_valid)
             return mean_absolute_error(y_valid, preds)
```

```
In [33]: print('Mean absolute error for RandomForest model is {}'.format(mae_rf(label_X_train,
```

```
Mean absolute error for RandomForest model is 1590.0152941176473
```

```
In [34]: X_test = pd.DataFrame({'Marketing Spend':[5000],'Visitors':[8000],'Day_Name':['Friday
```

```
In [35]: label_X_test = X_test.copy()
```

```
In [36]: label_X_test[['Promo','Day_Name']] = ordinal_encoder.transform(X_test[['Promo','Day_Na
```

```
In [37]:     model = RandomForestRegressor(n_estimators=200,random_state=0)
             model.fit(label_X_train, y_train)
             preds = model.predict(label_X_valid)
             print('Mean absolute error for RandomForest model is {}'.format(mean_absolute_erro
```

```
Mean absolute error for RandomForest model is 1590.0152941176473
```

```
In [38]: prediction = model.predict(label_X_test)
```

```
In [39]: print('Predicted revenue when Marketing spend is £5000, Visitors are 8000 for Promotic
```

```
Predicted revenue when Marketing spend is £5000, Visitors are 8000 for Promotion Red o
n Friday is [14045.83]
```

```
In [41]: X_test = pd.DataFrame({'Marketing Spend':[5000],'Visitors':[8000],'Day_Name':['Friday
         label_X_test = X_test.copy()
         label_X_test[['Promo','Day_Name']] = ordinal_encoder.transform(X_test[['Promo','Day_Na
         prediction = model.predict(label_X_test)
         print('Predicted revenue when Marketing spend is £5000, Visitors are 8000 for Promotic
```

```
Predicted revenue when Marketing spend is £5000, Visitors are 8000 for Promotion Blue
on Friday is [25037.07]
```

```
In [ ]:
```