```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sbn
from sklearn.model_selection import train_test_split
%matplotlib inline


plant1_gen = pd.read_csv('D:\sem5\de\mini_project\Plant_1_Generation_Data.csv')
plant2_gen = pd.read_csv('D:\sem5\de\mini_project\Plant_2_Generation_Data.csv')


plant1_sens = pd.read_csv('D:\sem5\de\mini_project\Plant_1_Weather_Sensor_Data.csv')
plant2_sens = pd.read_csv('D:\sem5\de\mini_project\Plant_2_Weather_Sensor_Data.csv')


plant1_gen
```

	DATE_TIME	PLANT_ID	SOURCE_KEY	DC_POWER	AC_POWER	DAILY_YIELD	TOTAL_YIELD
0	15-05-2020 00:00	4135001	1BY6WEcLGh8j5v7	0.0	0.0	0.000	6259559.0
1	15-05-2020 00:00	4135001	1IF53ai7Xc0U56Y	0.0	0.0	0.000	6183645.0
2	15-05-2020 00:00	4135001	3PZuoBAID5Wc2HD	0.0	0.0	0.000	6987759.0
3	15-05-2020 00:00	4135001	7JYdWkrLSPkdwr4	0.0	0.0	0.000	7602960.0
4	15-05-2020 00:00	4135001	McdE0feGgRqW7Ca	0.0	0.0	0.000	7158964.0
...	...	...	...	...	...	...	...
68773	17-06-2020 23:45	4135001	uHbuxQJl8lW7ozc	0.0	0.0	5967.000	7287002.0
68774	17-06-2020 23:45	4135001	wCURE6d3bPkepu2	0.0	0.0	5147.625	7028601.0
68775	17-06-2020 23:45	4135001	z9Y9gH1T5YWrNuG	0.0	0.0	5819.000	7251204.0
68776	17-06-2020 23:45	4135001	zBIq5rxdHJRwDNY	0.0	0.0	5817.000	6583369.0
68777	17-06-2020 23:45	4135001	zVJPv84UY57bAof	0.0	0.0	5910.000	7363272.0

68778 rows × 7 columns

```
plant2_gen
```

	DATE_TIME	PLANT_ID	SOURCE_KEY	DC_POWER	AC_POWER	DAILY_YIELD	TOTAL_YIELD
0	2020-05-15 00:00:00	4136001	4UPUqMRk7TRMgml	0.0	0.0	9425.000000	2.429011e+06
1	2020-05-15 00:00:00	4136001	81aHJ1q11NBPMrL	0.0	0.0	0.000000	1.215279e+09
2	2020-05-15 00:00:00	4136001	9kRcWv60rDACzjR	0.0	0.0	3075.333333	2.247720e+09
3	2020-05-15 00:00:00	4136001	Et9kgGMDl729KT4	0.0	0.0	269.933333	1.704250e+06
4	2020-05-15 00:00:00	4136001	IQ2d7wF4YD8zU1Q	0.0	0.0	3177.000000	1.994153e+07
...	...	...	...	...	...	...	...
67693	2020-06-17 23:45:00	4136001	q49J1IKaHRwDQnt	0.0	0.0	4157.000000	5.207580e+05
67694	2020-06-17 23:45:00	4136001	rrq4fwE8jgrTyWY	0.0	0.0	3931.000000	1.211314e+08
67695	2020-06-17 23:45:00	4136001	vOuJvMaM2sgwLmb	0.0	0.0	4322.000000	2.427691e+06
67696	2020-06-17 23:45:00	4136001	xMbIugepa2P7lBB	0.0	0.0	4218.000000	1.068964e+08
67697	2020-06-17 23:45:00	4136001	xoJJ8DcxJEcupym	0.0	0.0	4316.000000	2.093357e+08

67698 rows × 7 columns

Summation of yield

```
plant1_gendaily = plant1_gen.groupby('DATE_TIME')[['DC_POWER','AC_POWER', 'DAILY_YIELD','TOTAL_YIELD']].agg('sum').reset_index()
plant1_gendaily
```

	DATE_TIME	DC_POWER	AC_POWER	DAILY_YIELD	TOTAL_YIELD
0	01-06-2020 00:00	0.0	0.0	5407.250000	153519480.0
1	01-06-2020 00:15	0.0	0.0	0.000000	153519480.0
2	01-06-2020 00:30	0.0	0.0	0.000000	153519480.0
3	01-06-2020 00:45	0.0	0.0	0.000000	153519480.0
4	01-06-2020 01:00	0.0	0.0	0.000000	153519480.0
...	...	...	...	...	...
3153	31-05-2020 22:45	0.0	0.0	125291.000000	153519480.0
3154	31-05-2020 23:00	0.0	0.0	125291.000000	153519480.0
3155	31-05-2020 23:15	0.0	0.0	125291.000000	153519480.0
3156	31-05-2020 23:30	0.0	0.0	125291.000000	153519480.0
3157	31-05-2020 23:45	0.0	0.0	113737.142857	153519480.0

3158 rows × 5 columns

```
plant2_gendaily = plant2_gen.groupby('DATE_TIME')[['DC_POWER','AC_POWER', 'DAILY_YIELD','TOTAL_YIELD']].agg('sum').reset_index()
plant2_gendaily
```

	DATE_TIME	DC_POWER	AC_POWER	DAILY_YIELD	TOTAL_YIELD
0	2020-05-15 00:00:00	0.0	0.0	48899.938095	1.418960e+10
1	2020-05-15 00:15:00	0.0	0.0	28401.000000	1.418960e+10
2	2020-05-15 00:30:00	0.0	0.0	28401.000000	1.418960e+10
3	2020-05-15 00:45:00	0.0	0.0	28401.000000	1.418960e+10
4	2020-05-15 01:00:00	0.0	0.0	26516.000000	1.418960e+10
...	...	...	...	...	...
3254	2020-06-17 22:45:00	0.0	0.0	93040.000000	1.419408e+10
3255	2020-06-17 23:00:00	0.0	0.0	93040.000000	1.419408e+10
3256	2020-06-17 23:15:00	0.0	0.0	93040.000000	1.419408e+10
3257	2020-06-17 23:30:00	0.0	0.0	93040.000000	1.419408e+10
3258	2020-06-17 23:45:00	0.0	0.0	93040.000000	1.419408e+10

3259 rows × 5 columns

```
# Plant 1 generation data
plant1_gendaily['DATE_TIME'] = pd.to_datetime(plant1_gendaily['DATE_TIME'], format='%d-%m-%Y %H:%M')  # Convert to datetime with correct for
plant1_gendaily['TIME'] = plant1_gendaily['DATE_TIME'].dt.time  # Extract time from datetime
plant1_gendaily['DATE'] = plant1_gendaily['DATE_TIME'].dt.date  # Extract date from datetime

# Plant 2 generation data
plant2_gendaily['DATE_TIME'] = pd.to_datetime(plant2_gendaily['DATE_TIME'])  # Convert to datetime without specifying format (auto-infer)
plant2_gendaily['TIME'] = plant2_gendaily['DATE_TIME'].dt.time  # Extract time from datetime
plant2_gendaily['DATE'] = plant2_gendaily['DATE_TIME'].dt.date  # Extract date from datetime
```

```
plant1_gen.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 68778 entries, 0 to 68777
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   DATE_TIME    68778 non-null  object
 1   PLANT_ID     68778 non-null  int64
 2   SOURCE_KEY   68778 non-null  object
 3   DC_POWER     68778 non-null  float64
 4   AC_POWER     68778 non-null  float64
 5   DAILY_YIELD  68778 non-null  float64
 6   TOTAL_YIELD  68778 non-null  float64
dtypes: float64(4), int64(1), object(2)
memory usage: 3.7+ MB
```

```
plant2_gen.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 67698 entries, 0 to 67697
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   DATE_TIME    67698 non-null  object 
 1   PLANT_ID     67698 non-null  int64  
 2   SOURCE_KEY   67698 non-null  object 
 3   DC_POWER     67698 non-null  float64
 4   AC_POWER     67698 non-null  float64
 5   DAILY_YIELD  67698 non-null  float64
 6   TOTAL_YIELD  67698 non-null  float64
dtypes: float64(4), int64(1), object(2)
memory usage: 3.6+ MB
```

```
plant1_gendaily.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3158 entries, 0 to 3157
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype         
---  ------       --------------  -----         
 0   DATE_TIME    3158 non-null   datetime64[ns]
 1   DC_POWER     3158 non-null   float64       
 2   AC_POWER     3158 non-null   float64       
 3   DAILY_YIELD  3158 non-null   float64       
 4   TOTAL_YIELD  3158 non-null   float64       
 5   TIME         3158 non-null   object        
 6   DATE         3158 non-null   object        
dtypes: datetime64[ns](1), float64(4), object(2)
memory usage: 172.8+ KB
```

```
plant1_gendaily.describe()
```

	DATE_TIME	DC_POWER	AC_POWER	DAILY_YIELD	TOTAL_YIELD
count	3158	3158.000000	3158.000000	3158.000000	3.158000e+03
mean	2020-06-01 06:42:44.344521728	68547.713729	6703.628149	71782.817545	1.519892e+08
min	2020-05-15 00:00:00	0.000000	0.000000	0.000000	2.654004e+07
25%	2020-05-23 23:18:45	0.000000	0.000000	90.750000	1.520976e+08
50%	2020-06-01 12:37:30	8515.285714	823.033036	66068.000000	1.535320e+08
75%	2020-06-09 17:56:15	140386.504463	13750.606696	129398.500000	1.549950e+08
max	2020-06-17 23:45:00	298937.785710	29150.212499	193770.000000	1.561428e+08
std	NaN	88044.612181	8603.120476	65974.417997	1.061670e+07

```
plant2_gendaily.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3259 entries, 0 to 3258
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype         
---  ------       --------------  -----         
 0   DATE_TIME    3259 non-null   datetime64[ns]
 1   DC_POWER     3259 non-null   float64       
 2   AC_POWER     3259 non-null   float64       
 3   DAILY_YIELD  3259 non-null   float64       
 4   TOTAL_YIELD  3259 non-null   float64       
 5   TIME         3259 non-null   object        
 6   DATE         3259 non-null   object        
dtypes: datetime64[ns](1), float64(4), object(2)
memory usage: 178.4+ KB
```

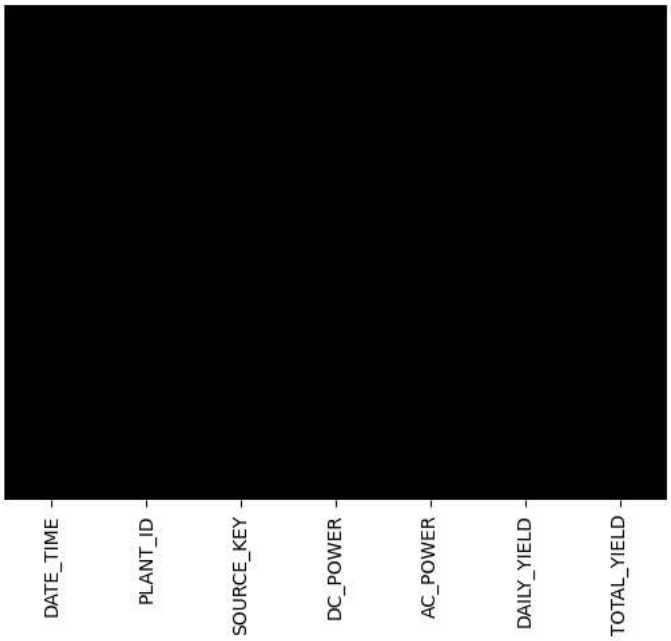
```
plant2_gendaily.describe()
```

	DATE_TIME	DC_POWER	AC_POWER	DAILY_YIELD	TOTAL_YIELD
count	3259	3259.000000	3259.000000	3259.000000	3.259000e+03
mean	2020-06-01 00:04:35.053697536	5124.648465	5011.974903	68443.535809	1.368802e+10
min	2020-05-15 00:00:00	0.000000	0.000000	0.000000	0.000000e+00
25%	2020-05-23 12:07:30	0.000000	0.000000	18698.245238	1.335913e+10
50%	2020-06-01 00:00:00	494.427143	477.536667	73875.000000	1.419016e+10
75%	2020-06-09 12:07:30	11048.773333	10795.727619	110975.223810	1.419312e+10
max	2020-06-17 23:45:00	26630.506667	25979.760476	162876.000000	1.419408e+10
std	NaN	6462.118509	6317.872611	48505.077129	1.245968e+09

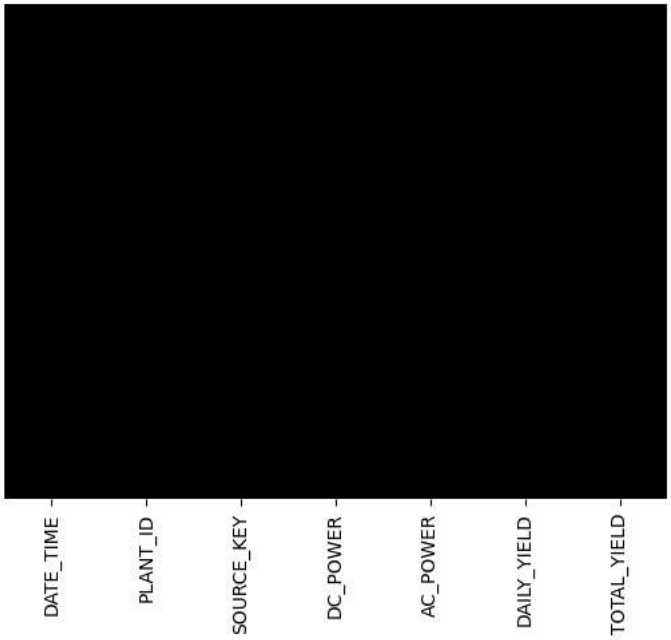
Missing values

```
print('Plant 1 Generation Data')
sbn.heatmap(plant1_gen.isnull(), yticklabels=False, cbar=False, cmap='magma')
plt.show()

print('Plant 2 Generation Data')
sbn.heatmap(plant2_gen.isnull(), yticklabels=False, cbar=False, cmap='magma')
plt.show()
```

Plant 1 Generation Data



Plant 2 Generation Data

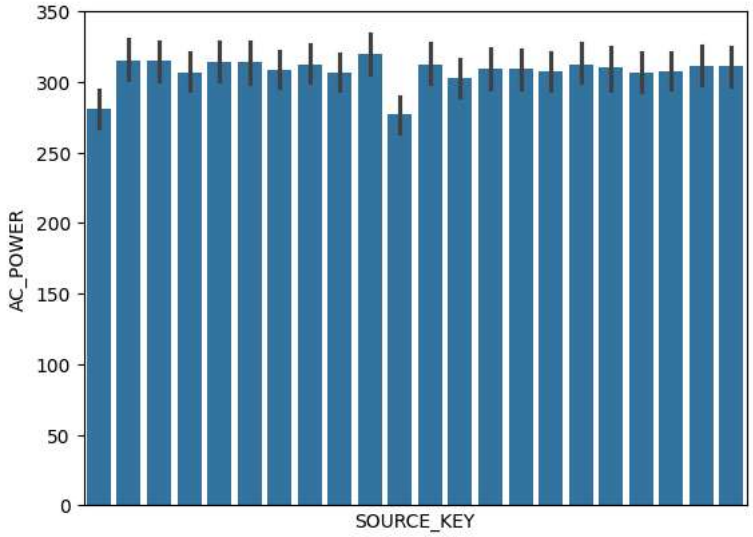



```
print('There are {} inverters in Solar Power Plant 1.'.format(plant1_gen['SOURCE_KEY'].nunique()))
print('There are {} inverters in Solar Power Plant 2.'.format(plant2_gen['SOURCE_KEY'].nunique()))
```

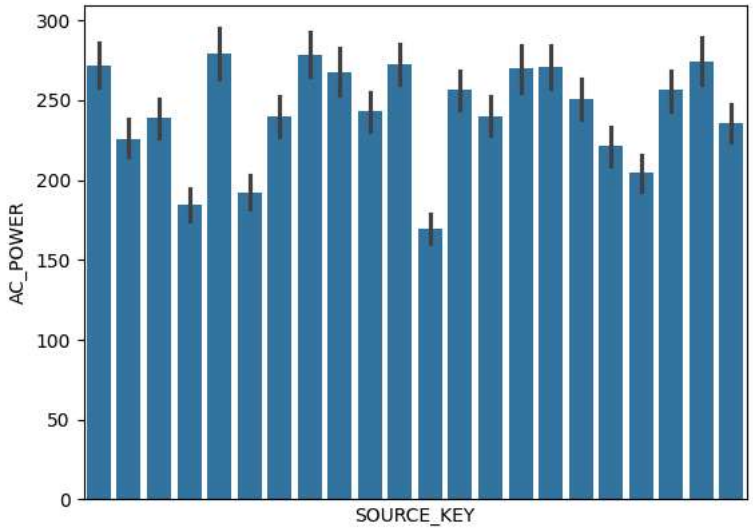
```
There are 22 inverters in Solar Power Plant 1.
There are 22 inverters in Solar Power Plant 2.
```

```
print('Plant 1 Inverters')
sbn.barplot(x='SOURCE_KEY', y='AC_POWER', data=plant1_gen)
plt.xticks([])
plt.show()

print('Plant 2 Inverters')
sbn.barplot(x='SOURCE_KEY', y='AC_POWER', data=plant2_gen)
plt.xticks([])
plt.show()
```

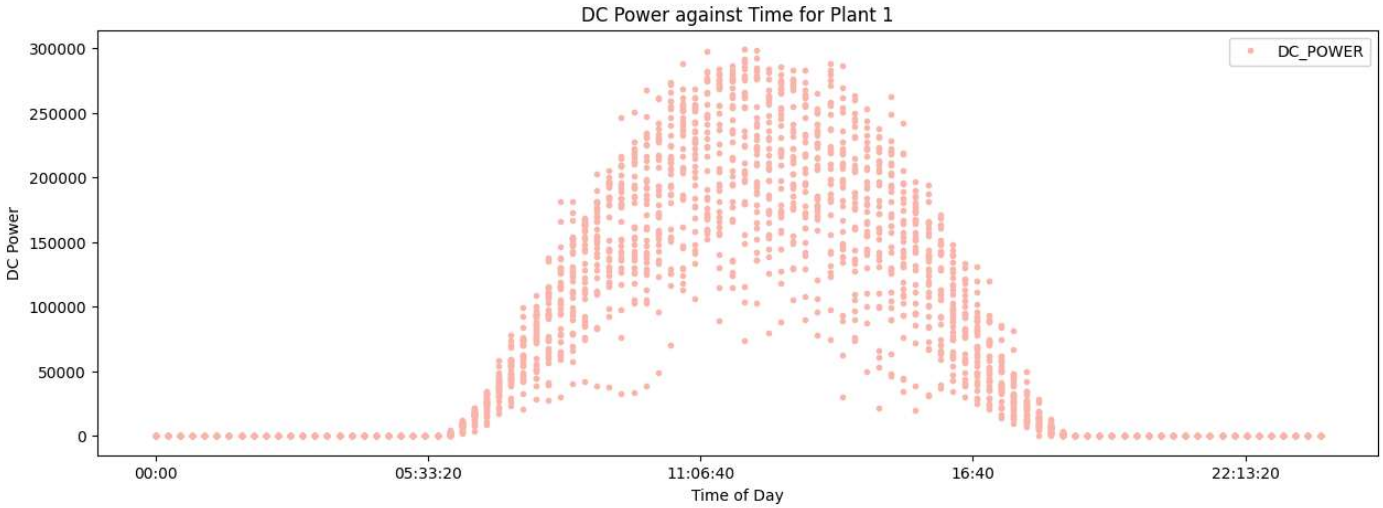
Plant 1 Inverters

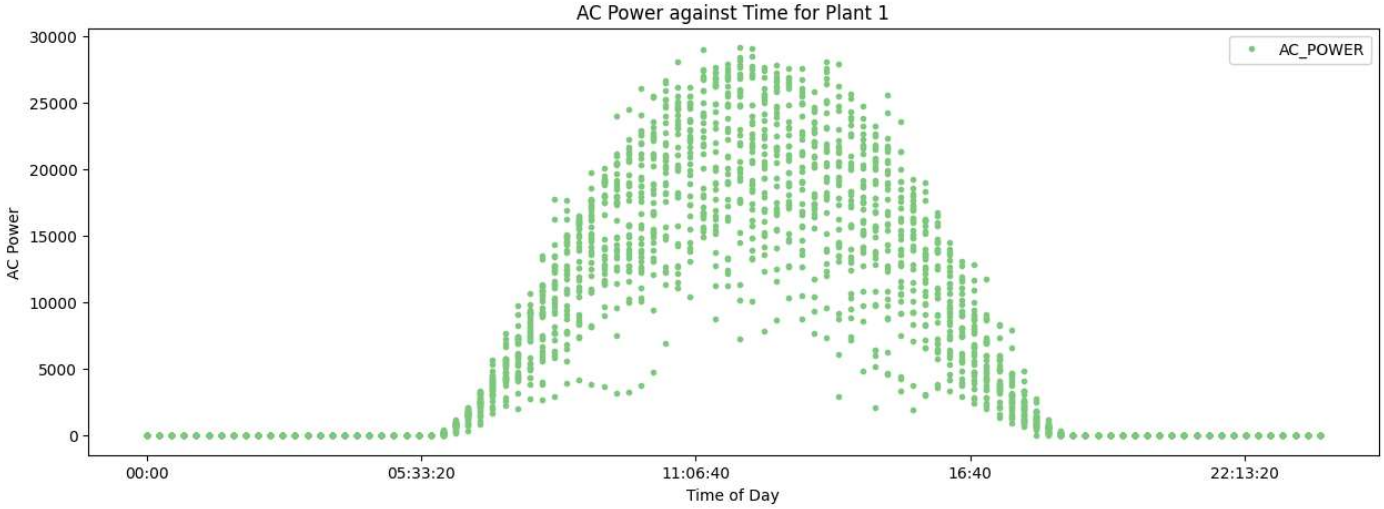


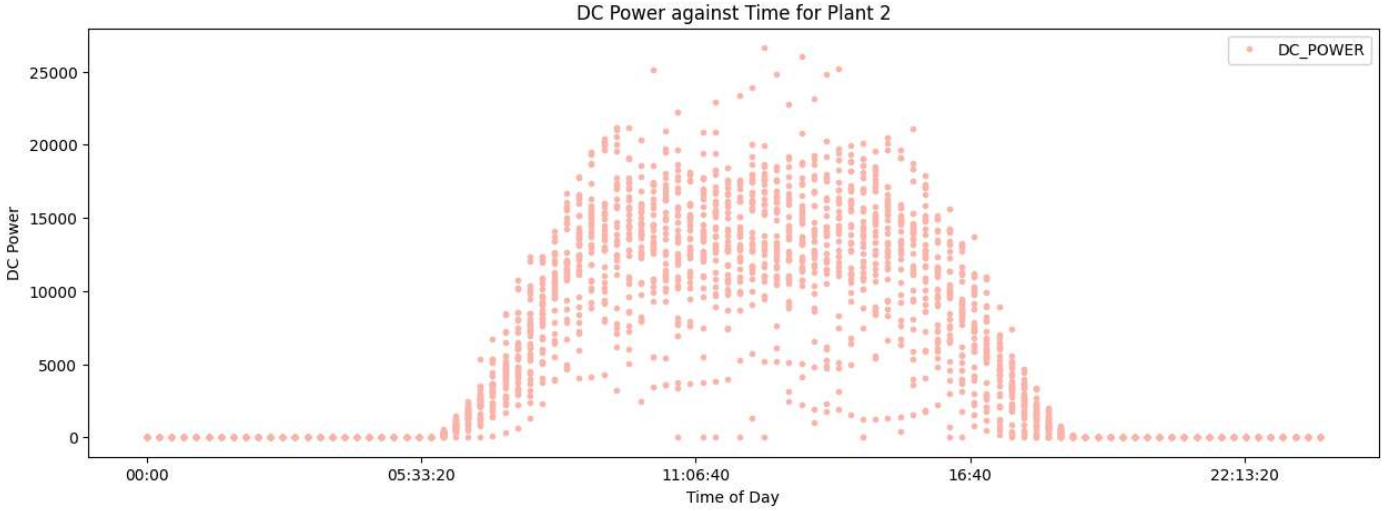
Plant 2 Inverters

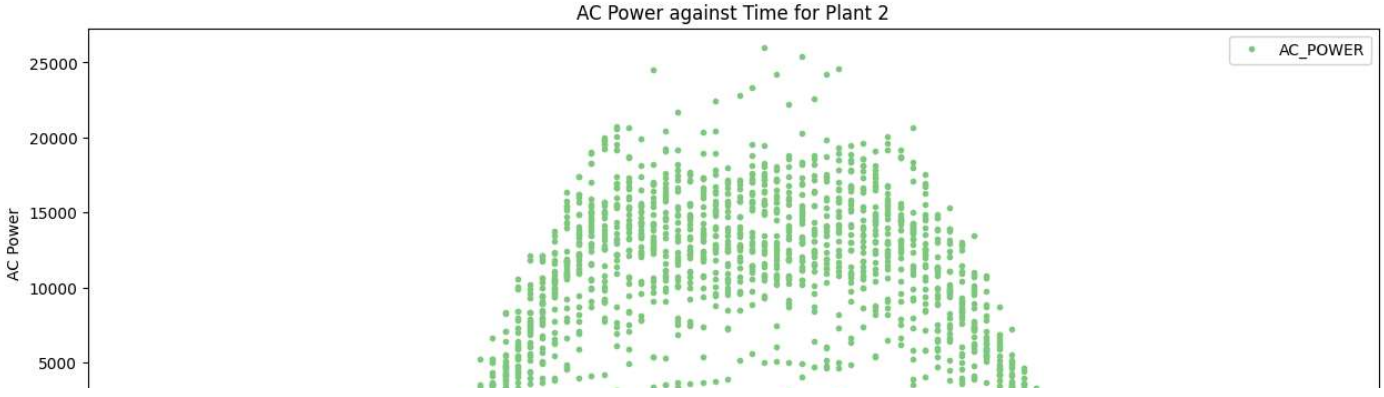


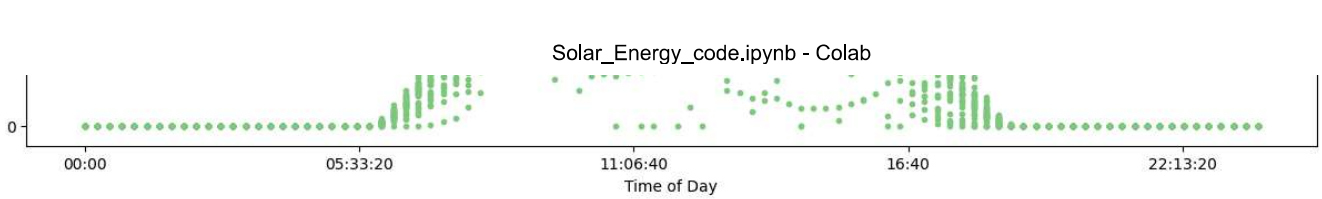

```
plant1_gendaily.plot(x= 'TIME', y='DC_POWER', style='.', figsize = (15, 5), colormap='Pastel1')
plt.ylabel('DC Power')
plt.xlabel('Time of Day')
plt.title('DC Power against Time for Plant 1')
plt.show()

plant1_gendaily.plot(x= 'TIME', y='AC_POWER', style='.', figsize = (15, 5), colormap='Accent')
plt.ylabel('AC Power')
plt.xlabel('Time of Day')
plt.title('AC Power against Time for Plant 1')
plt.show()

plant2_gendaily.plot(x= 'TIME', y='DC_POWER', style='.', figsize = (15, 5), colormap='Pastel1')
plt.ylabel('DC Power')
plt.xlabel('Time of Day')
plt.title('DC Power against Time for Plant 2')
plt.show()

plant2_gendaily.plot(x= 'TIME', y='AC_POWER', style='.', figsize = (15, 5), colormap='Accent')
plt.ylabel('AC Power')
plt.xlabel('Time of Day')
plt.title('AC Power against Time for Plant 2')
plt.show()
```


DC Power against Time for Plant 1
DC_POWER
DC Power
Time of Day
00:00
05:33:20
11:06:40
16:40
22:13:20


AC Power against Time for Plant 1
AC_POWER
AC Power
Time of Day
00:00
05:33:20
11:06:40
16:40
22:13:20

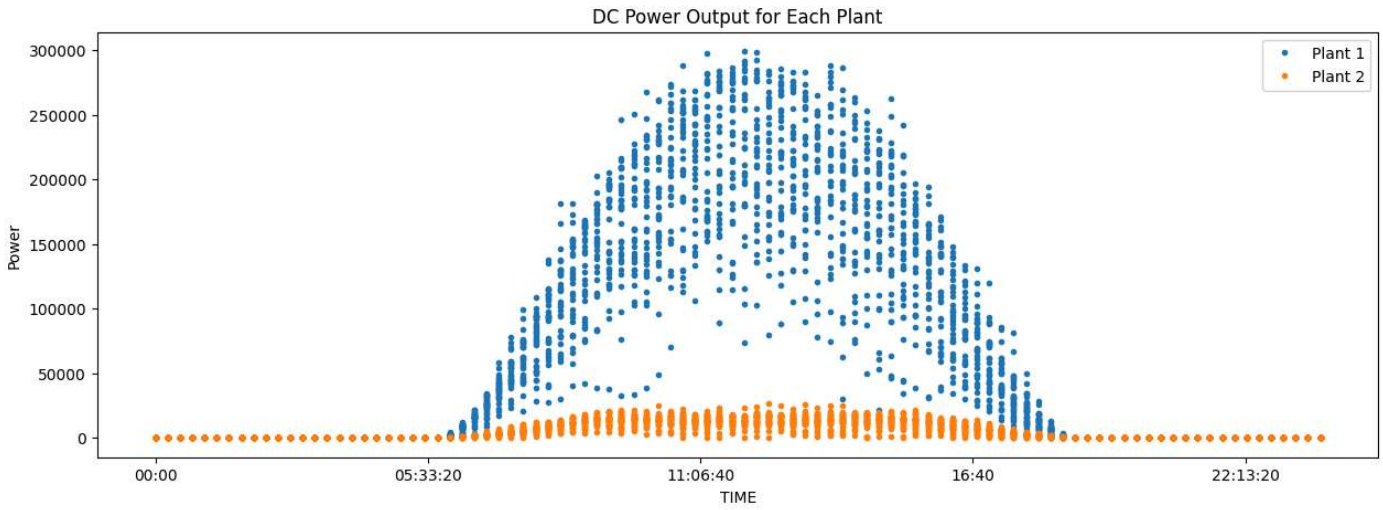

DC Power against Time for Plant 2
DC_POWER
DC Power
Time of Day
00:00
05:33:20
11:06:40
16:40
22:13:20

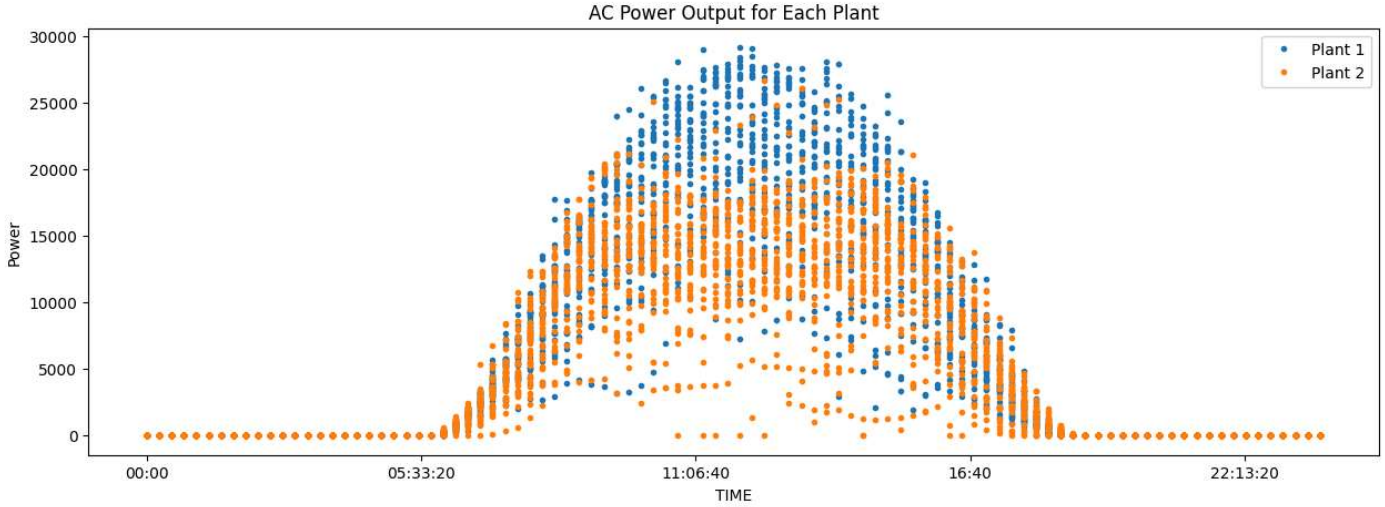

AC Power against Time for Plant 2
AC_POWER
AC Power




```
DCcompare = plant1_gendaily.plot(x='TIME', y='DC_POWER', figsize=(15,5), legend=True, style='.', label='Plant 1')
plant2_gendaily.plot(x='TIME', y='DC_POWER', legend=True, style='.', label='Plant 2', ax=DCcompare)
plt.title('DC Power Output for Each Plant')
plt.ylabel('Power')
plt.show()
```

```
ACcompare = plant1_gendaily.plot(x='TIME', y='AC_POWER', figsize=(15,5), legend=True, style='.', label='Plant 1')
plant2_gendaily.plot(x='TIME', y='DC_POWER', legend=True, style='.', label='Plant 2', ax=ACcompare)
plt.title('AC Power Output for Each Plant')
plt.ylabel('Power')
plt.show()
```



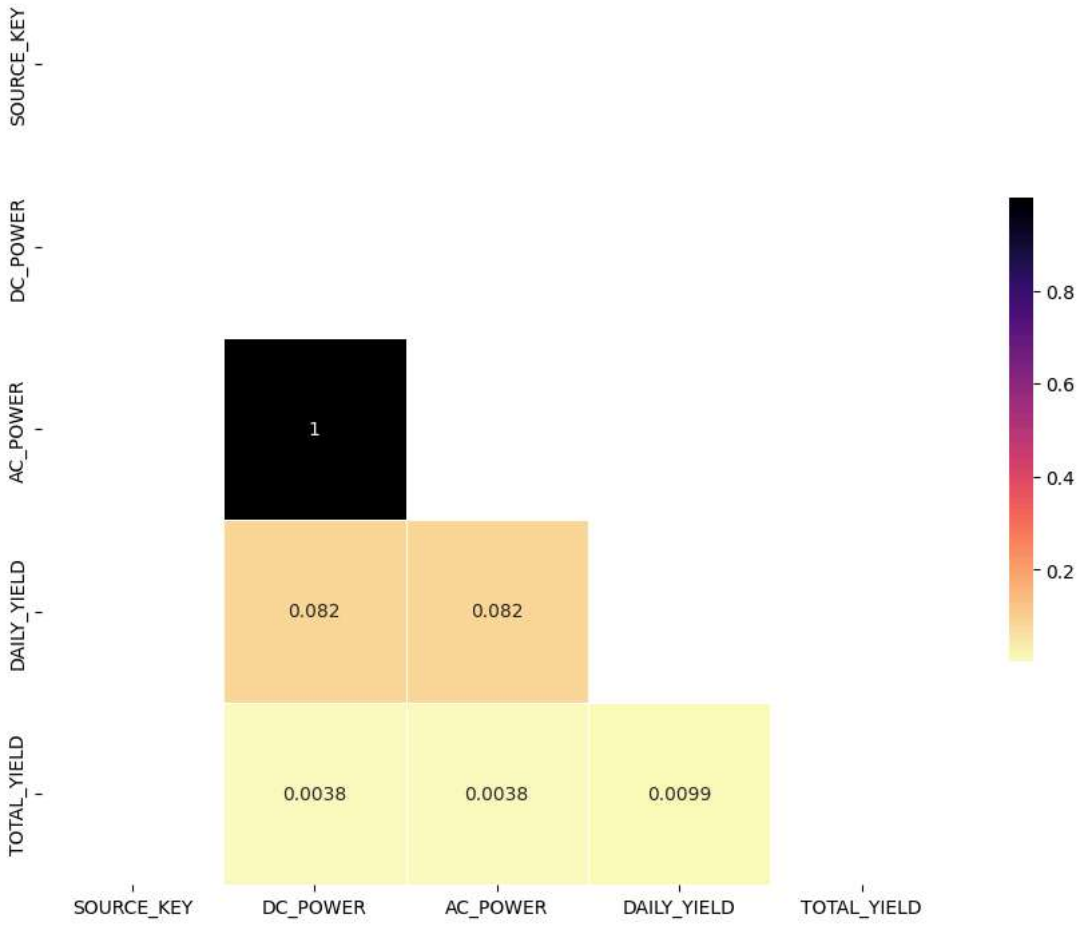


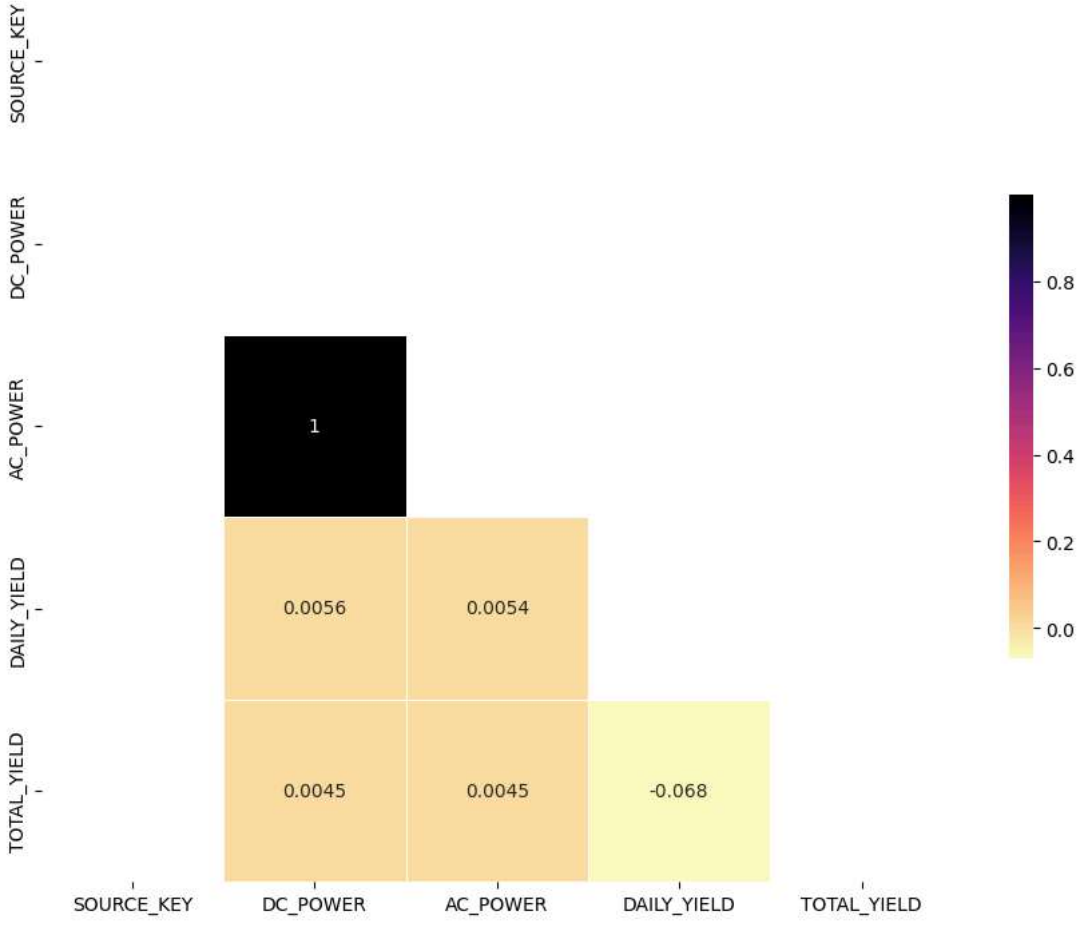

```
# Drop non-numeric columns like 'PLANT_ID' and 'DATE_TIME' for correlation calculation
g1corr = plant1_gen.drop(['PLANT_ID', 'DATE_TIME'], axis=1).apply(pd.to_numeric, errors='coerce').corr()
g2corr = plant2_gen.drop(['PLANT_ID', 'DATE_TIME'], axis=1).apply(pd.to_numeric, errors='coerce').corr()

# Print the correlation matrices
print('Plant 1 Generation Data Correlation Coefficient')
print(g1corr)

print('Plant 2 Generation Data Correlation Coefficient')
print(g2corr)
```

```
Plant 1 Generation Data Correlation Coefficient
             SOURCE_KEY  DC_POWER  AC_POWER  DAILY_YIELD  TOTAL_YIELD
SOURCE_KEY          NaN       NaN       NaN          NaN          NaN
DC_POWER            NaN  1.000000  0.999996     0.082284     0.003815
AC_POWER            NaN  0.999996  1.000000     0.082234     0.003804
DAILY_YIELD         NaN  0.082284  0.082234     1.000000     0.009867
TOTAL_YIELD         NaN  0.003815  0.003804     0.009867     1.000000
Plant 2 Generation Data Correlation Coefficient
             SOURCE_KEY  DC_POWER  AC_POWER  DAILY_YIELD  TOTAL_YIELD
SOURCE_KEY          NaN       NaN       NaN          NaN          NaN
DC_POWER            NaN  1.000000  0.999997     0.005593     0.004528
AC_POWER            NaN  0.999997  1.000000     0.005395     0.004533
DAILY_YIELD         NaN  0.005593  0.005395     1.000000    -0.068472
TOTAL_YIELD         NaN  0.004528  0.004533    -0.068472     1.000000
```

```
g1mask = np.triu(np.ones_like(g1corr, dtype=bool))
g2mask = np.triu(np.ones_like(g2corr, dtype=bool))

cmap = 'magma_r'

f, ax = plt.subplots(figsize=(11, 9))
sbn.heatmap(g1corr, mask=g1mask, cmap=cmap, square=True, linewidths=.5, cbar_kws={"shrink": .5}, annot=True)
plt.show()

f, ax = plt.subplots(figsize=(11, 9))
sbn.heatmap(g2corr, mask=g2mask, cmap=cmap, square=True, linewidths=.5, cbar_kws={"shrink": .5}, annot=True)
plt.show()
```

[+]


SOURCE_KEY
DC_POWER
AC_POWER
DAILY_YIELD
TOTAL_YIELD
1
0.082
0.082
0.0038
0.0038
0.0099
0.8
0.6
0.4
0.2


SOURCE_KEY
DC_POWER
AC_POWER
DAILY_YIELD
TOTAL_YIELD
1
0.0056
0.0054
0.0045
0.0045
-0.068
0.8
0.6
0.4
0.2
0.0

```
plant1_sens
```

	DATE_TIME	PLANT_ID	SOURCE_KEY	AMBIENT_TEMPERATURE	MODULE_TEMPERATURE	IRRADIATION
0	2020-05-15 00:00:00	4135001	HmiyD2TTLFNqkNe	25.184316	22.857507	0.0
1	2020-05-15 00:15:00	4135001	HmiyD2TTLFNqkNe	25.084589	22.761668	0.0
2	2020-05-15 00:30:00	4135001	HmiyD2TTLFNqkNe	24.935753	22.592306	0.0
3	2020-05-15 00:45:00	4135001	HmiyD2TTLFNqkNe	24.846130	22.360852	0.0
4	2020-05-15 01:00:00	4135001	HmiyD2TTLFNqkNe	24.621525	22.165423	0.0
...	...	...	...	...	...	...
3177	2020-06-17 22:45:00	4135001	HmiyD2TTLFNqkNe	22.150570	21.480377	0.0
3178	2020-06-17 23:00:00	4135001	HmiyD2TTLFNqkNe	22.129816	21.389024	0.0
3179	2020-06-17 23:15:00	4135001	HmiyD2TTLFNqkNe	22.008275	20.709211	0.0
3180	2020-06-17 23:30:00	4135001	HmiyD2TTLFNqkNe	21.969495	20.734963	0.0
3181	2020-06-17 23:45:00	4135001	HmiyD2TTLFNqkNe	21.909288	20.427972	0.0

3182 rows × 6 columns

```
plant1_sens
```

	DATE_TIME	PLANT_ID	SOURCE_KEY	AMBIENT_TEMPERATURE	MODULE_TEMPERATURE	IRRADIATION
0	2020-05-15 00:00:00	4135001	HmiyD2TTLFNqkNe	25.184316	22.857507	0.0
1	2020-05-15 00:15:00	4135001	HmiyD2TTLFNqkNe	25.084589	22.761668	0.0
2	2020-05-15 00:30:00	4135001	HmiyD2TTLFNqkNe	24.935753	22.592306	0.0
3	2020-05-15 00:45:00	4135001	HmiyD2TTLFNqkNe	24.846130	22.360852	0.0
4	2020-05-15 01:00:00	4135001	HmiyD2TTLFNqkNe	24.621525	22.165423	0.0
...	...	...	...	...	...	...
3177	2020-06-17 22:45:00	4135001	HmiyD2TTLFNqkNe	22.150570	21.480377	0.0
3178	2020-06-17 23:00:00	4135001	HmiyD2TTLFNqkNe	22.129816	21.389024	0.0
3179	2020-06-17 23:15:00	4135001	HmiyD2TTLFNqkNe	22.008275	20.709211	0.0
3180	2020-06-17 23:30:00	4135001	HmiyD2TTLFNqkNe	21.969495	20.734963	0.0
3181	2020-06-17 23:45:00	4135001	HmiyD2TTLFNqkNe	21.909288	20.427972	0.0

3182 rows × 6 columns

```
plant1_sensdaily = plant1_sens.groupby('DATE_TIME')[['AMBIENT_TEMPERATURE','MODULE_TEMPERATURE',
                                                     'IRRADIATION', 'SOURCE_KEY']].agg('sum').reset_index()
```

```
plant1_sensdaily
```

	DATE_TIME	AMBIENT_TEMPERATURE	MODULE_TEMPERATURE	IRRADIATION	SOURCE_KEY
0	2020-05-15 00:00:00	25.184316	22.857507	0.0	HmiyD2TTLFNqkNe
1	2020-05-15 00:15:00	25.084589	22.761668	0.0	HmiyD2TTLFNqkNe
2	2020-05-15 00:30:00	24.935753	22.592306	0.0	HmiyD2TTLFNqkNe
3	2020-05-15 00:45:00	24.846130	22.360852	0.0	HmiyD2TTLFNqkNe
4	2020-05-15 01:00:00	24.621525	22.165423	0.0	HmiyD2TTLFNqkNe
...	...	...	...	...	...
3177	2020-06-17 22:45:00	22.150570	21.480377	0.0	HmiyD2TTLFNqkNe
3178	2020-06-17 23:00:00	22.129816	21.389024	0.0	HmiyD2TTLFNqkNe
3179	2020-06-17 23:15:00	22.008275	20.709211	0.0	HmiyD2TTLFNqkNe
3180	2020-06-17 23:30:00	21.969495	20.734963	0.0	HmiyD2TTLFNqkNe
3181	2020-06-17 23:45:00	21.909288	20.427972	0.0	HmiyD2TTLFNqkNe

3182 rows × 5 columns

```
plant2_sensdaily = plant2_sens.groupby('DATE_TIME')[['AMBIENT_TEMPERATURE','MODULE_TEMPERATURE',
                                                     'IRRADIATION', 'SOURCE_KEY']].agg('sum').reset_index()
```

```
plant2_sensdaily
```

	DATE_TIME	AMBIENT_TEMPERATURE	MODULE_TEMPERATURE	IRRADIATION	SOURCE_KEY
0	2020-05-15 00:00:00	27.004764	25.060789	0.0	iq8k7ZNt4Mwm3w0
1	2020-05-15 00:15:00	26.880811	24.421869	0.0	iq8k7ZNt4Mwm3w0
2	2020-05-15 00:30:00	26.682055	24.427290	0.0	iq8k7ZNt4Mwm3w0
3	2020-05-15 00:45:00	26.500589	24.420678	0.0	iq8k7ZNt4Mwm3w0
4	2020-05-15 01:00:00	26.596148	25.088210	0.0	iq8k7ZNt4Mwm3w0
...	...	...	...	...	...
3254	2020-06-17 22:45:00	23.511703	22.856201	0.0	iq8k7ZNt4Mwm3w0
3255	2020-06-17 23:00:00	23.482282	22.744190	0.0	iq8k7ZNt4Mwm3w0
3256	2020-06-17 23:15:00	23.354743	22.492245	0.0	iq8k7ZNt4Mwm3w0
3257	2020-06-17 23:30:00	23.291048	22.373909	0.0	iq8k7ZNt4Mwm3w0
3258	2020-06-17 23:45:00	23.202871	22.535908	0.0	iq8k7ZNt4Mwm3w0

3259 rows × 5 columns

```
#Plant 1 sensor data
plant1_sensdaily['DATE_TIME'] = pd.to_datetime(plant1_sensdaily['DATE_TIME'])
plant1_sensdaily['TIME'] = plant1_sensdaily['DATE_TIME'].dt.time
plant1_sensdaily['DATE'] = pd.to_datetime(plant1_sensdaily['DATE_TIME'].dt.date)

#Plant 2 sensor data
plant2_sensdaily['DATE_TIME'] = pd.to_datetime(plant2_sensdaily['DATE_TIME'])
plant2_sensdaily['TIME'] = plant2_sensdaily['DATE_TIME'].dt.time
plant2_sensdaily['DATE'] = pd.to_datetime(plant2_sensdaily['DATE_TIME'].dt.date)
```

```
plant1_sensdaily.info()
```

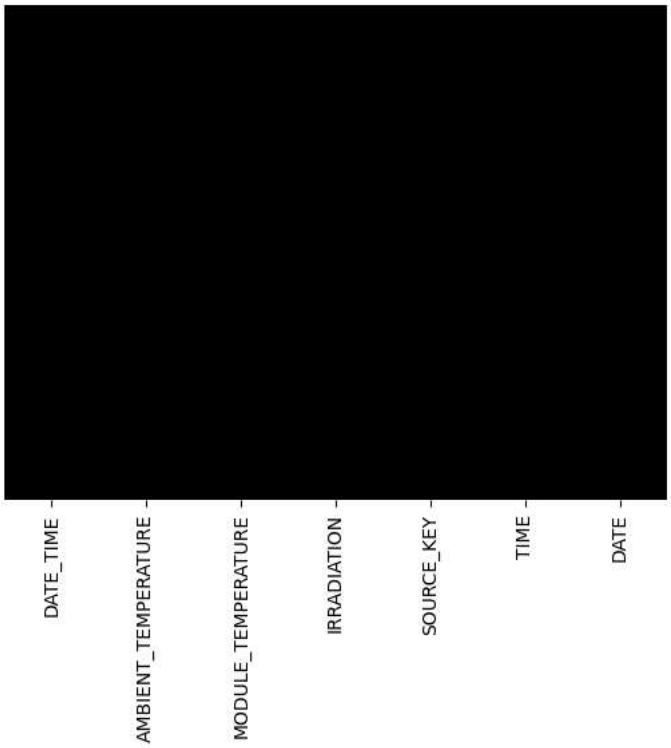
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3182 entries, 0 to 3181
Data columns (total 7 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   DATE_TIME            3182 non-null   datetime64[ns]
 1   AMBIENT_TEMPERATURE  3182 non-null   float64
 2   MODULE_TEMPERATURE   3182 non-null   float64
 3   IRRADIATION          3182 non-null   float64
 4   SOURCE_KEY           3182 non-null   object
 5   TIME                 3182 non-null   object
 6   DATE                 3182 non-null   datetime64[ns]
dtypes: datetime64[ns](2), float64(3), object(2)
memory usage: 174.1+ KB
```

```
plant2_sensdaily.info()
```

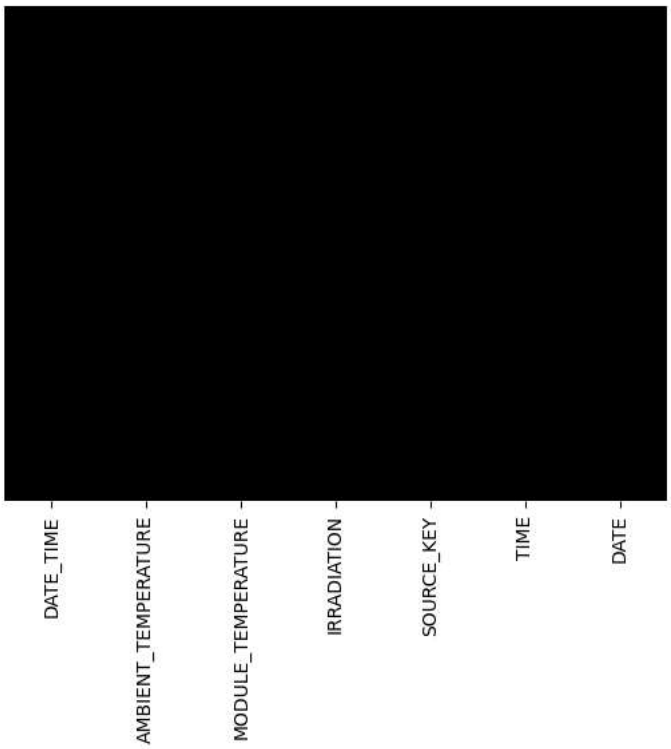
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3259 entries, 0 to 3258
Data columns (total 7 columns):
 #   Column               Non-Null Count  Dtype         
---  ------               --------------  -----         
 0   DATE_TIME            3259 non-null   datetime64[ns]
 1   AMBIENT_TEMPERATURE  3259 non-null   float64       
 2   MODULE_TEMPERATURE   3259 non-null   float64       
 3   IRRADIATION          3259 non-null   float64       
 4   SOURCE_KEY           3259 non-null   object        
 5   TIME                 3259 non-null   object        
 6   DATE                 3259 non-null   datetime64[ns]
dtypes: datetime64[ns](2), float64(3), object(2)
memory usage: 178.4+ KB
```

```
print('Plant 1 Sensor Data')
sbn.heatmap(plant1_sensdaily.isnull(), yticklabels=False, cbar=False, cmap='magma')
plt.show()

print('Plant 2 Sensor Data')
sbn.heatmap(plant2_sensdaily.isnull(), yticklabels=False, cbar=False, cmap='magma')
plt.show()
```

Plant 1 Sensor Data



Plant 2 Sensor Data

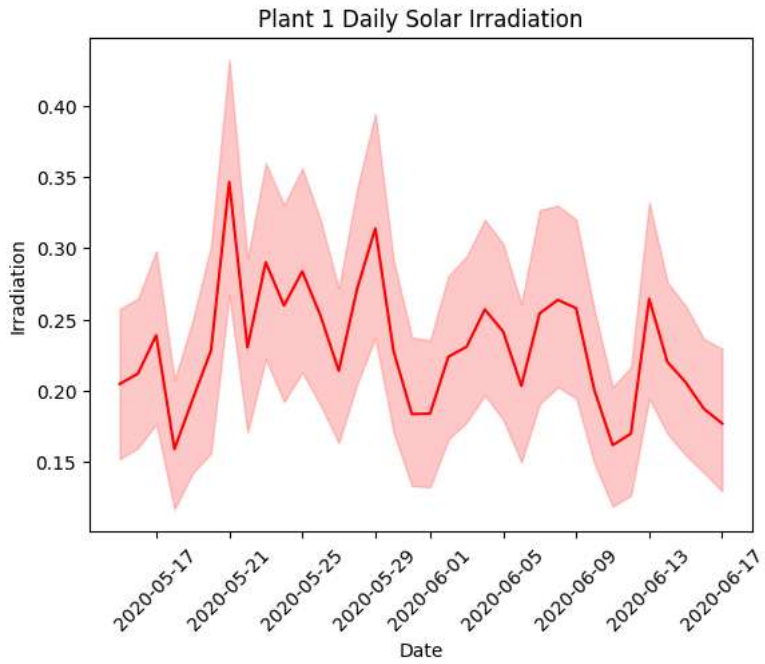


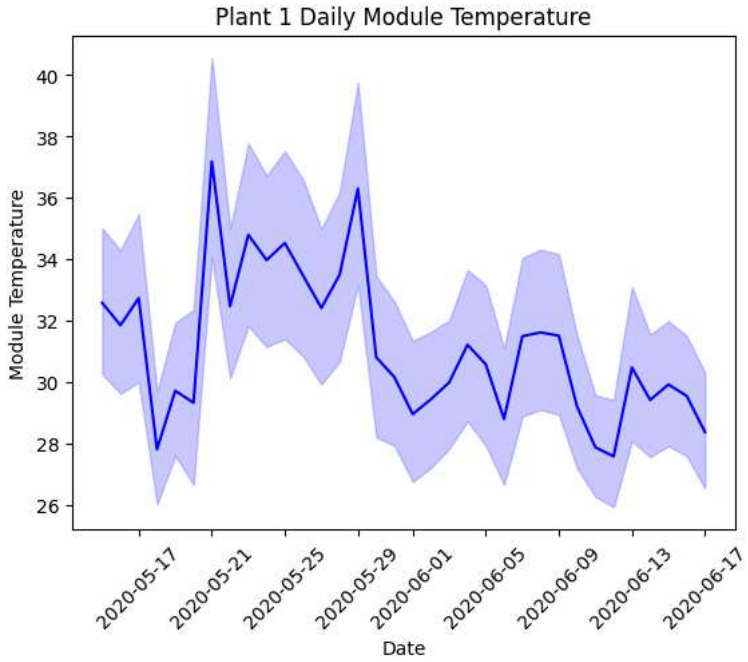

```
plant1_sensdaily['DATE'].nunique()
```

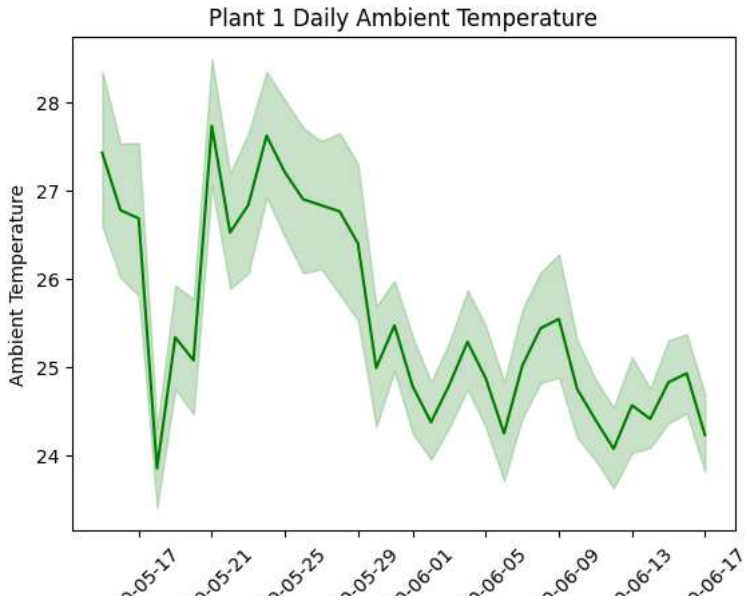
```
34
```

```
#Plant 1
#Irradiation
sbn.lineplot(x='DATE', y='IRRADIATION', data=plant1_sensdaily, err_style='band', color='red')

plt.ylabel('Irradiation')
plt.xlabel('Date')
plt.title('Plant 1 Daily Solar Irradiation')
plt.xticks(rotation=45)
plt.show()
```

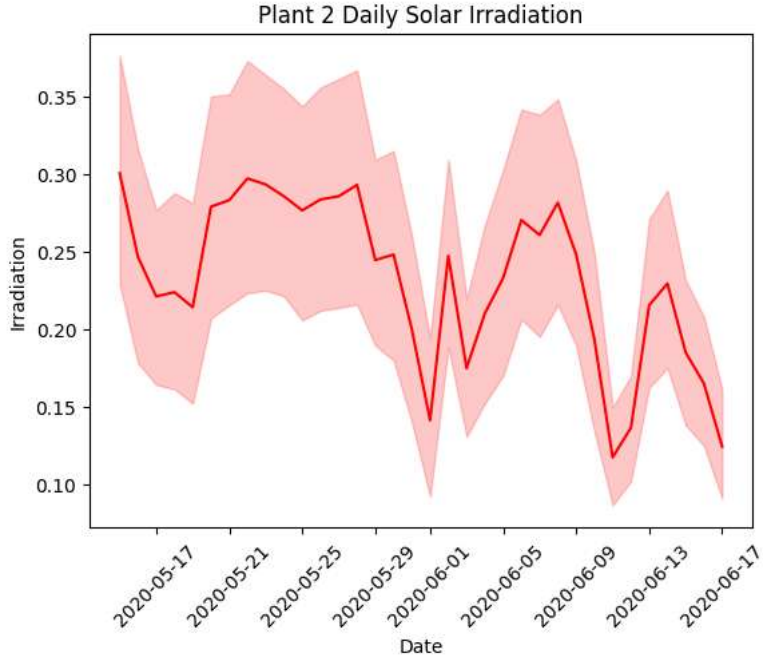
```
#Module Temperature
sbn.lineplot(x='DATE', y='MODULE_TEMPERATURE', data=plant1_sensdaily, err_style='band', color='blue')

plt.ylabel('Module Temperature')
plt.xlabel('Date')
plt.title('Plant 1 Daily Module Temperature')
plt.xticks(rotation=45)
plt.show()

#Ambient Temperature
sbn.lineplot(x='DATE', y='AMBIENT_TEMPERATURE', data=plant1_sensdaily, err_style='band', color='green')

plt.ylabel('Ambient Temperature')
plt.xlabel('Date')
plt.title('Plant 1 Daily Ambient Temperature')
plt.xticks(rotation=45)
plt.show()

#Plant 2
#Irradiation
sbn.lineplot(x='DATE', y='IRRADIATION', data=plant2_sensdaily, err_style='band', color='red')

plt.ylabel('Irradiation')
plt.xlabel('Date')
plt.title('Plant 2 Daily Solar Irradiation')
plt.xticks(rotation=45)
plt.show()

#Module Temperature
sbn.lineplot(x='DATE', y='MODULE_TEMPERATURE', data=plant2_sensdaily, err_style='band', color='blue')

plt.ylabel('Module Temperature')
plt.xlabel('Date')
plt.title('Plant 2 Daily Module Temperature')
plt.xticks(rotation=45)
plt.show()

#Ambient Temperature
sbn.lineplot(x='DATE', y='AMBIENT_TEMPERATURE', data=plant2_sensdaily, err_style='band', color='green')

plt.ylabel('Ambient Temperature')
plt.xlabel('Date')
plt.title('Plant 2 Daily Ambient Temperature')
plt.xticks(rotation=45)
plt.show()
```

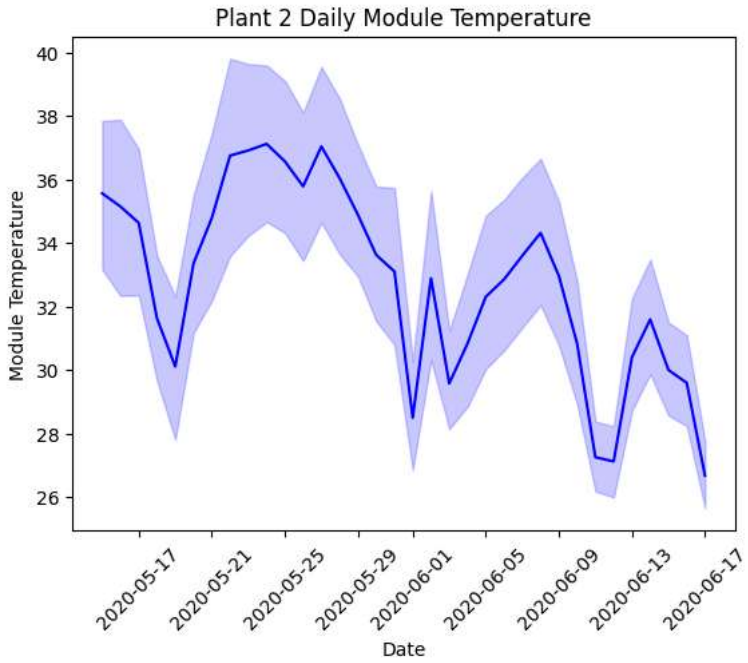

Plant 1 Daily Solar Irradiation
Irradiation
0.40
0.35
0.30
0.25
0.20
0.15
2020-05-17
2020-05-21
2020-05-25
2020-05-29
2020-06-01
2020-06-05
2020-06-09
2020-06-13
2020-06-17
Date

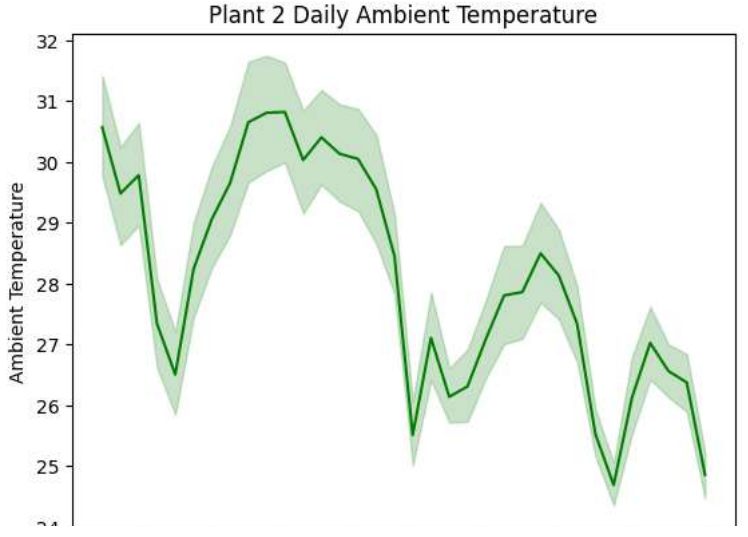

Plant 1 Daily Module Temperature
Module Temperature
40
38
36
34
32
30
28
26
2020-05-17
2020-05-21
2020-05-25
2020-05-29
2020-06-01
2020-06-05
2020-06-09
2020-06-13
2020-06-17
Date

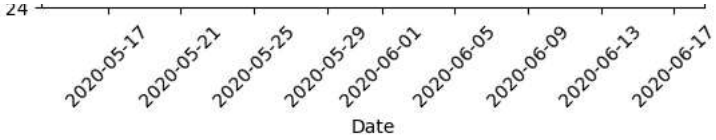

Plant 1 Daily Ambient Temperature
Ambient Temperature
28
27
26
25
24


2020
2020
2020
2020
2020
2020
2020
2020
2020
Date

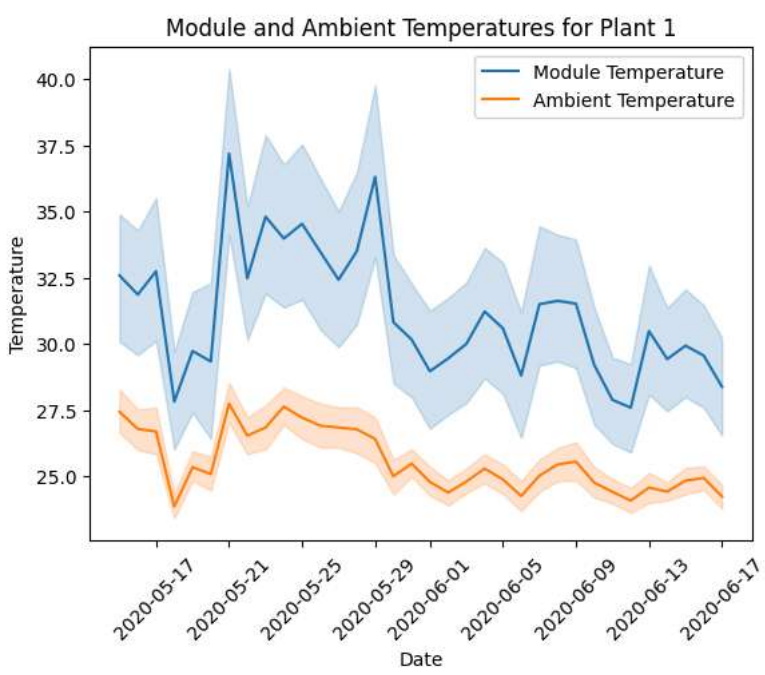

Plant 2 Daily Solar Irradiation
Irradiation
0.35
0.30
0.25
0.20
0.15
0.10
2020-05-17
2020-05-21
2020-05-25
2020-05-29
2020-06-01
2020-06-05
2020-06-09
2020-06-13
2020-06-17
Date

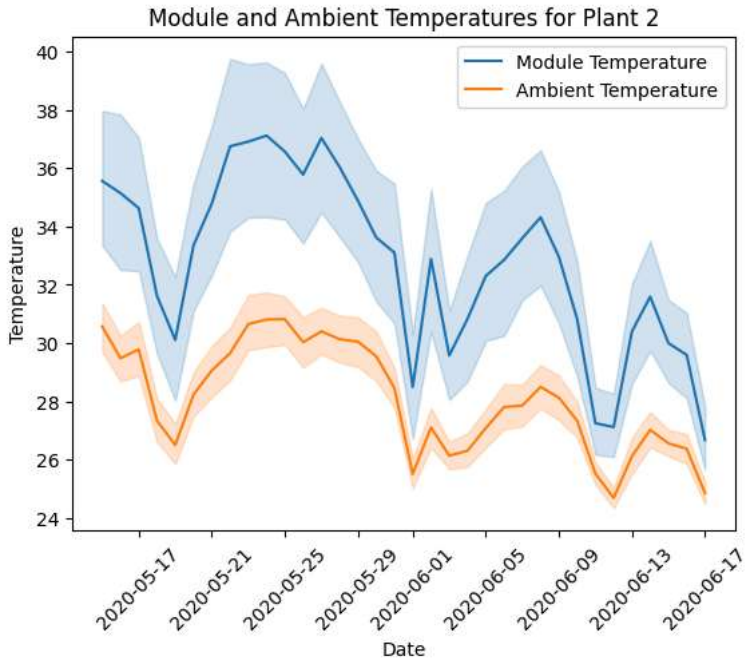

Plant 2 Daily Module Temperature
Module Temperature
40
38
36
34
32
30
28
26
2020-05-17
2020-05-21
2020-05-25
2020-05-29
2020-06-01
2020-06-05
2020-06-09
2020-06-13
2020-06-17
Date


Plant 2 Daily Ambient Temperature
Ambient Temperature
32
31
30
29
28
27
26
25


24
2020-05-17
2020-05-21
2020-05-25
2020-05-29
2020-06-01
2020-06-05
2020-06-09
2020-06-13
2020-06-17
Date

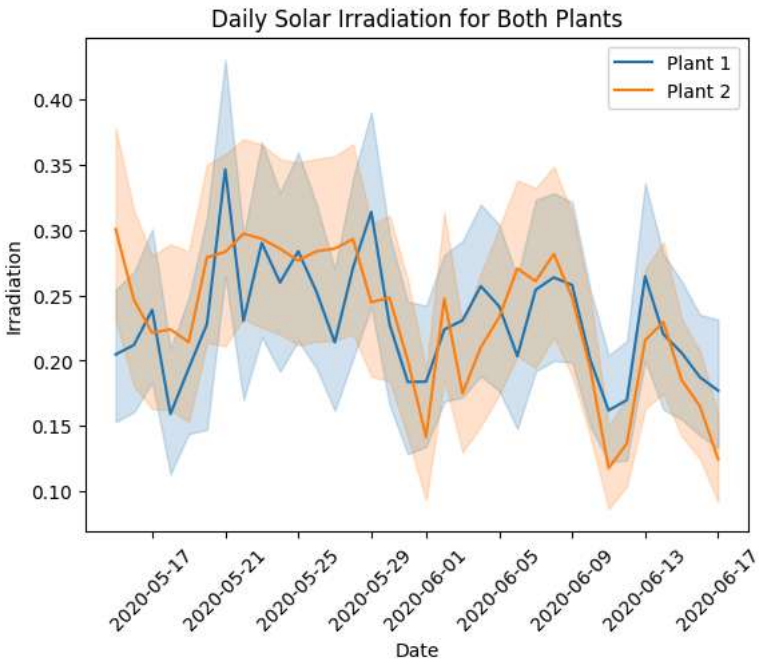
```
tempsens_plant1 = sbn.lineplot(x='DATE', y='MODULE_TEMPERATURE', data=plant1_sensdaily, err_style='band', label='Module Temperature')
sbn.lineplot(x='DATE', y='AMBIENT_TEMPERATURE', data=plant1_sensdaily, err_style='band', label='Ambient Temperature', ax=tempsens_plant1)
plt.ylabel('Temperature')
plt.xlabel('Date')
plt.title('Module and Ambient Temperatures for Plant 1')
plt.xticks(rotation=45)
plt.show()

tempsens_plant2 = sbn.lineplot(x='DATE', y='MODULE_TEMPERATURE', data=plant2_sensdaily, err_style='band', label='Module Temperature')
sbn.lineplot(x='DATE', y='AMBIENT_TEMPERATURE', data=plant2_sensdaily, err_style='band', label='Ambient Temperature', ax=tempsens_plant2)
plt.ylabel('Temperature')
plt.xlabel('Date')
plt.title('Module and Ambient Temperatures for Plant 2')
plt.xticks(rotation=45)
plt.show()
```

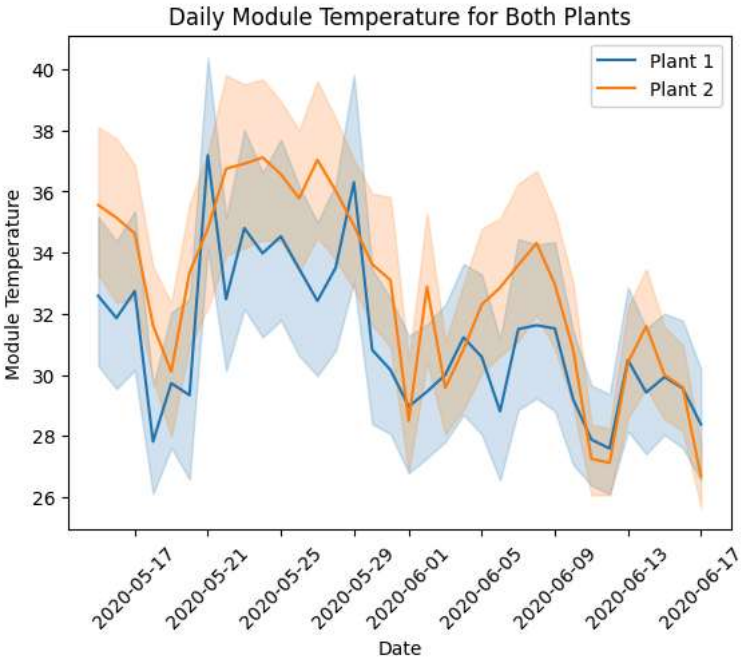


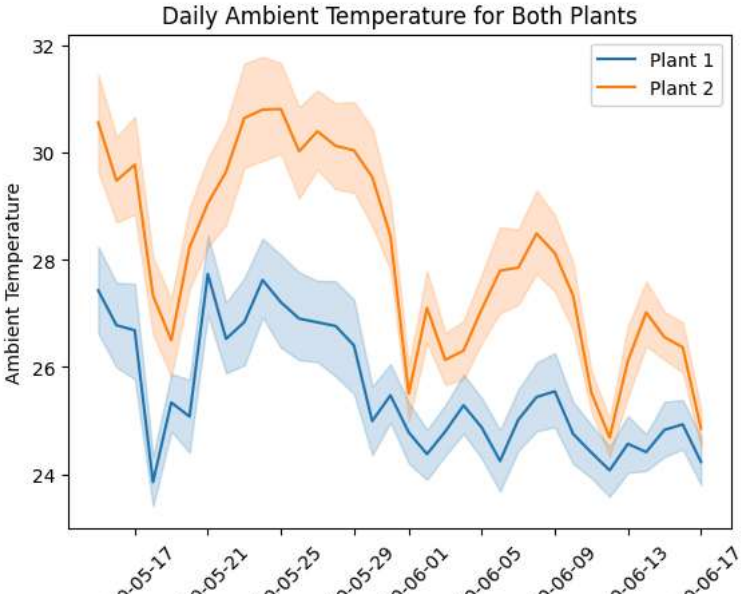



```
# Daily Irradiation
ambient_compare = sbn.lineplot(x='DATE', y='IRRADIATION', data=plant1_sensdaily, err_style='band', label='Plant 1')
sbn.lineplot(x='DATE', y='IRRADIATION', data=plant2_sensdaily, err_style='band', label='Plant 2', ax=ambient_compare)
plt.ylabel('Irradiation')
plt.xlabel('Date')
```

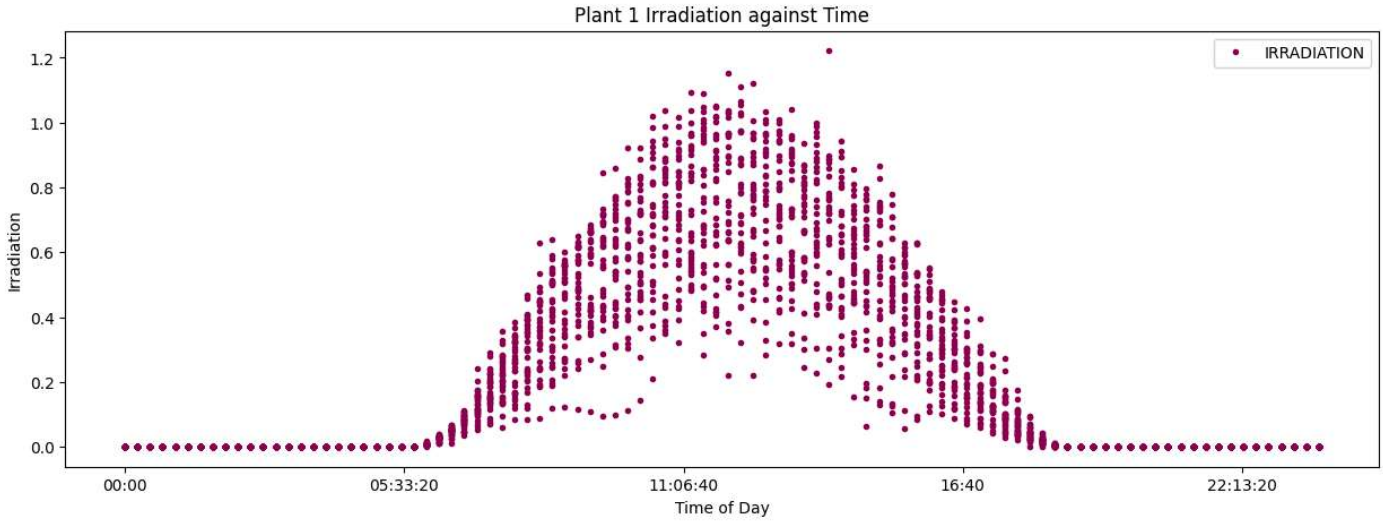
```
plt.title('Daily Solar Irradiation for Both Plants')
plt.xticks(rotation=45)
plt.show()

# Daily Module Temperature
modtemp_compare = sbn.lineplot(x='DATE', y='MODULE_TEMPERATURE', data=plant1_sensdaily, err_style='band', label='Plant 1')
sbn.lineplot(x='DATE', y='MODULE_TEMPERATURE', data=plant2_sensdaily, err_style='band', label='Plant 2', ax=modtemp_compare)
plt.ylabel('Module Temperature')
plt.xlabel('Date')
plt.title('Daily Module Temperature for Both Plants')
plt.xticks(rotation=45)
plt.show()

# Daily Ambient Temperature
ambtemp_compare = sbn.lineplot(x='DATE', y='AMBIENT_TEMPERATURE', data=plant1_sensdaily, err_style='band', label='Plant 1')
sbn.lineplot(x='DATE', y='AMBIENT_TEMPERATURE', data=plant2_sensdaily, err_style='band', label='Plant 2', ax=ambtemp_compare)
plt.ylabel('Ambient Temperature')
plt.xlabel('Date')
plt.title('Daily Ambient Temperature for Both Plants')
plt.xticks(rotation=45)
plt.show()
```

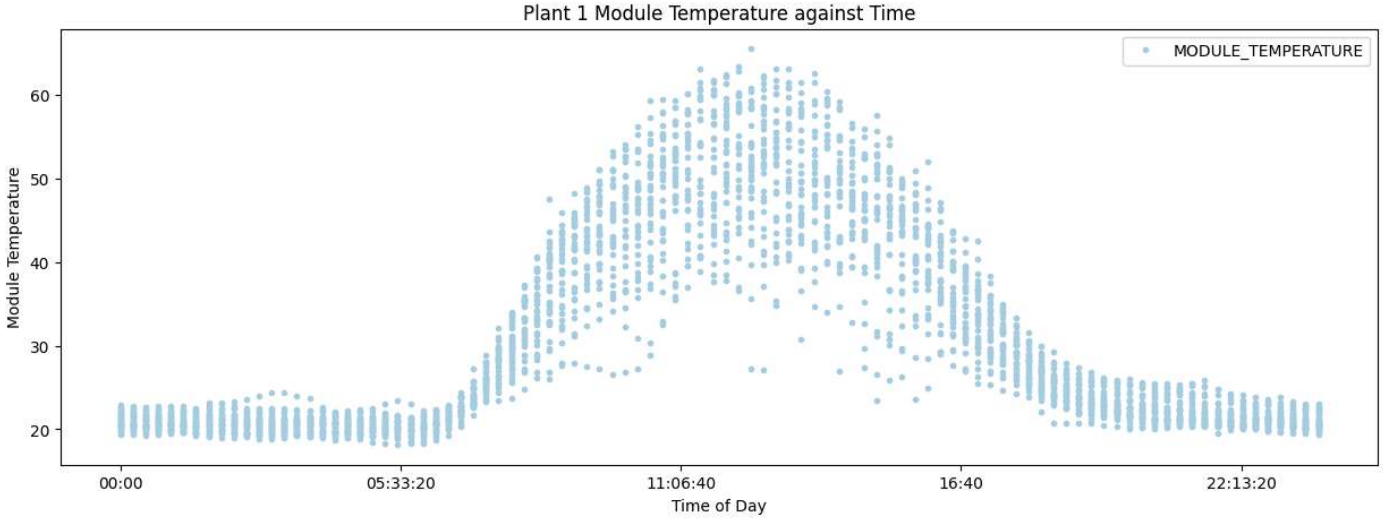

Daily Solar Irradiation for Both Plants
Plant 1
Plant 2
Irradiation
0.40
0.35
0.30
0.25
0.20
0.15
0.10
2020-05-17
2020-05-21
2020-05-25
2020-05-29
2020-06-01
2020-06-05
2020-06-09
2020-06-13
2020-06-17
Date

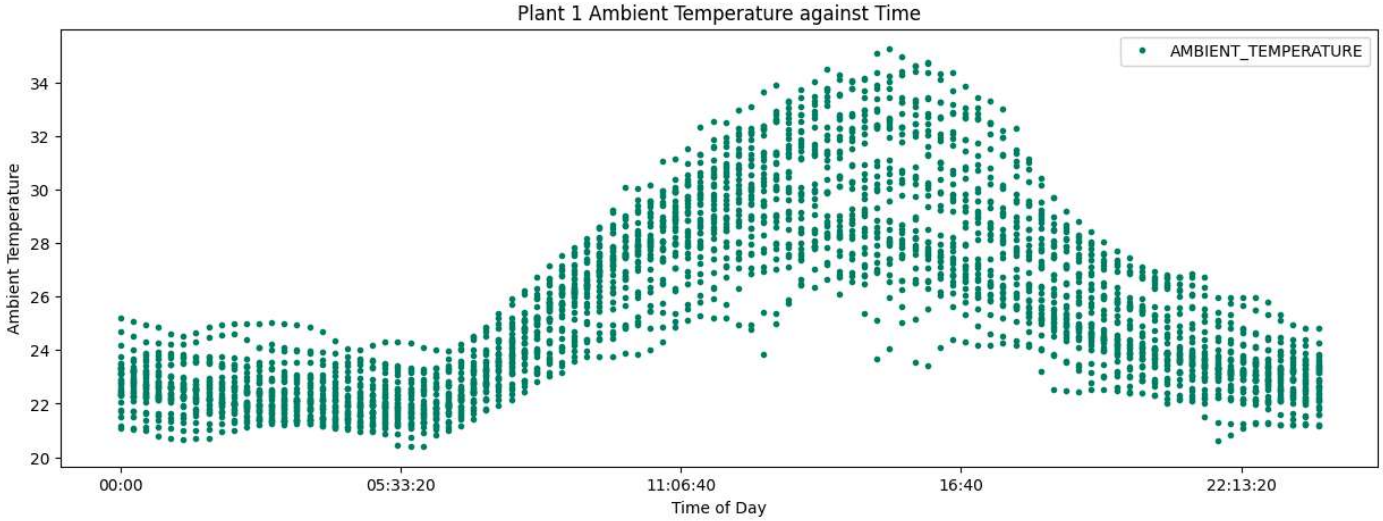

Daily Module Temperature for Both Plants
Plant 1
Plant 2
Module Temperature
40
38
36
34
32
30
28
26
2020-05-17
2020-05-21
2020-05-25
2020-05-29
2020-06-01
2020-06-05
2020-06-09
2020-06-13
2020-06-17
Date

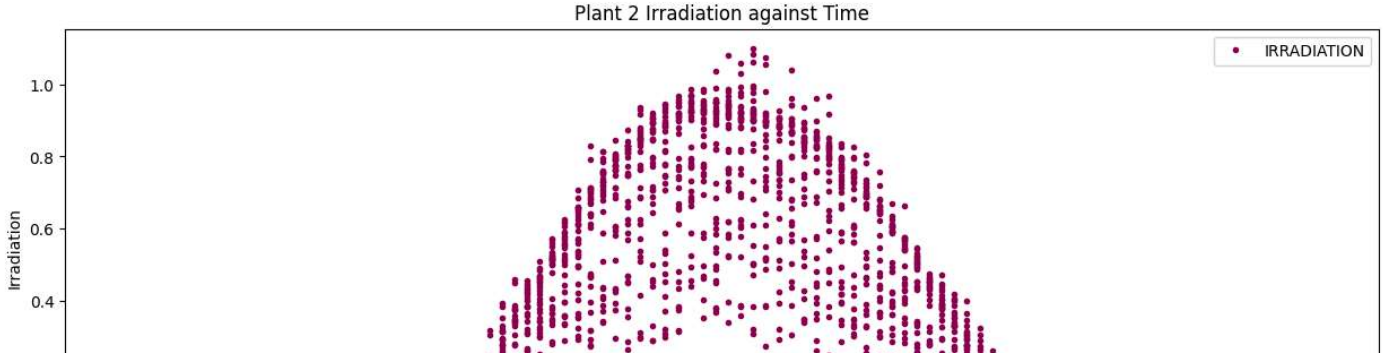

Daily Ambient Temperature for Both Plants
Plant 1
Plant 2
Ambient Temperature
32
30
28
26
24

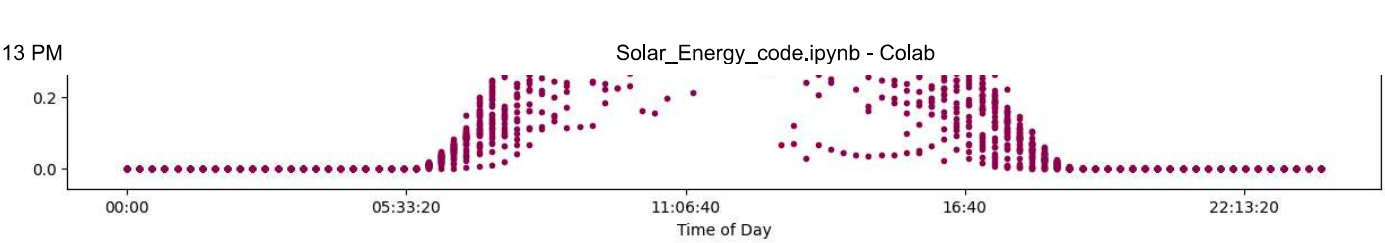

```
# Plant 1
# Irradiation
plant1_sensdaily.plot(x= 'TIME', y='IRRADIATION', style='.', figsize = (15, 5), colormap='PiYG')
plt.ylabel('Irradiation')
plt.xlabel('Time of Day')
plt.title('Plant 1 Irradiation against Time')
plt.show()

# Module Temperature
plant1_sensdaily.plot(x= 'TIME', y='MODULE_TEMPERATURE', style='.', figsize = (15, 5), colormap='Paired')
plt.ylabel('Module Temperature')
plt.xlabel('Time of Day')
plt.title('Plant 1 Module Temperature against Time')
plt.show()

# Ambient Temperature
plant1_sensdaily.plot(x= 'TIME', y='AMBIENT_TEMPERATURE', style='.', figsize = (15, 5), colormap='summer')
plt.ylabel('Ambient Temperature')
plt.xlabel('Time of Day')
plt.title('Plant 1 Ambient Temperature against Time')
plt.show()

# Plant 2
# Irradiation
plant2_sensdaily.plot(x= 'TIME', y='IRRADIATION', style='.', figsize = (15, 5), colormap='PiYG')
plt.ylabel('Irradiation')
plt.xlabel('Time of Day')
plt.title('Plant 2 Irradiation against Time')
plt.show()

# Module Temperature
plant2_sensdaily.plot(x= 'TIME', y='MODULE_TEMPERATURE', style='.', figsize = (15, 5), colormap='Paired')
plt.ylabel('Module Temperature')
plt.xlabel('Time of Day')
plt.title('Plant 2 Module Temperature against Time')
plt.show()

# Ambient Temperature
plant2_sensdaily.plot(x= 'TIME', y='AMBIENT_TEMPERATURE', style='.', figsize = (15, 5), colormap='summer')
plt.ylabel('Ambient Temperature')
plt.xlabel('Time of Day')
plt.title('Plant 2 Ambient Temperature against Time')
plt.show()
```

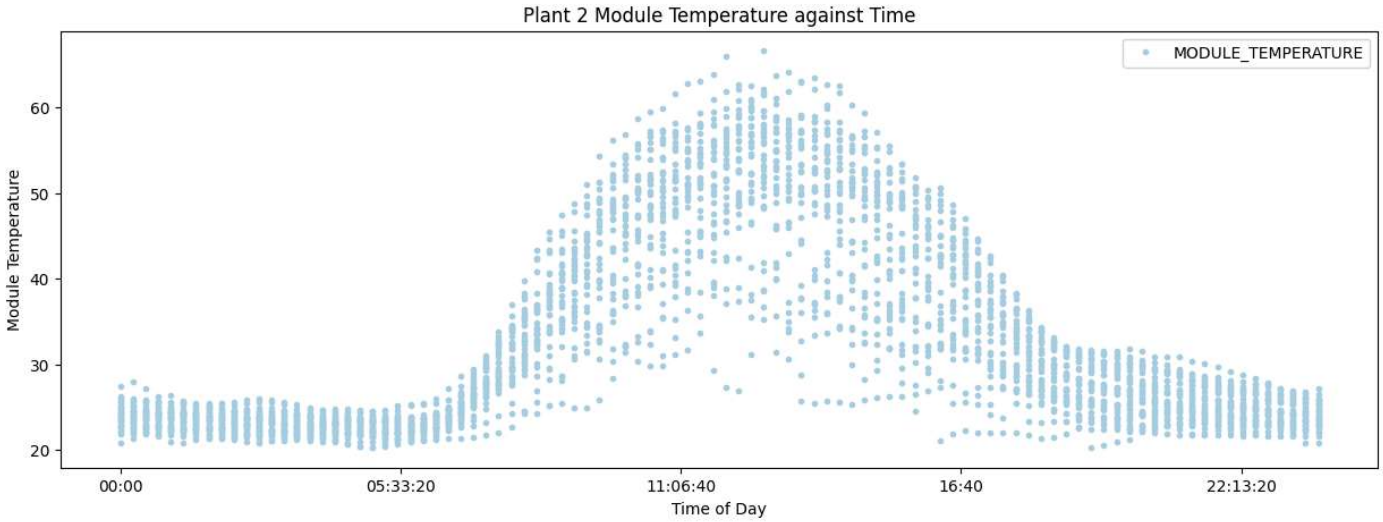

Plant 1 Irradiation against Time
IRRADIATION
Irradiation
Time of Day
00:00
05:33:20
11:06:40
16:40
22:13:20

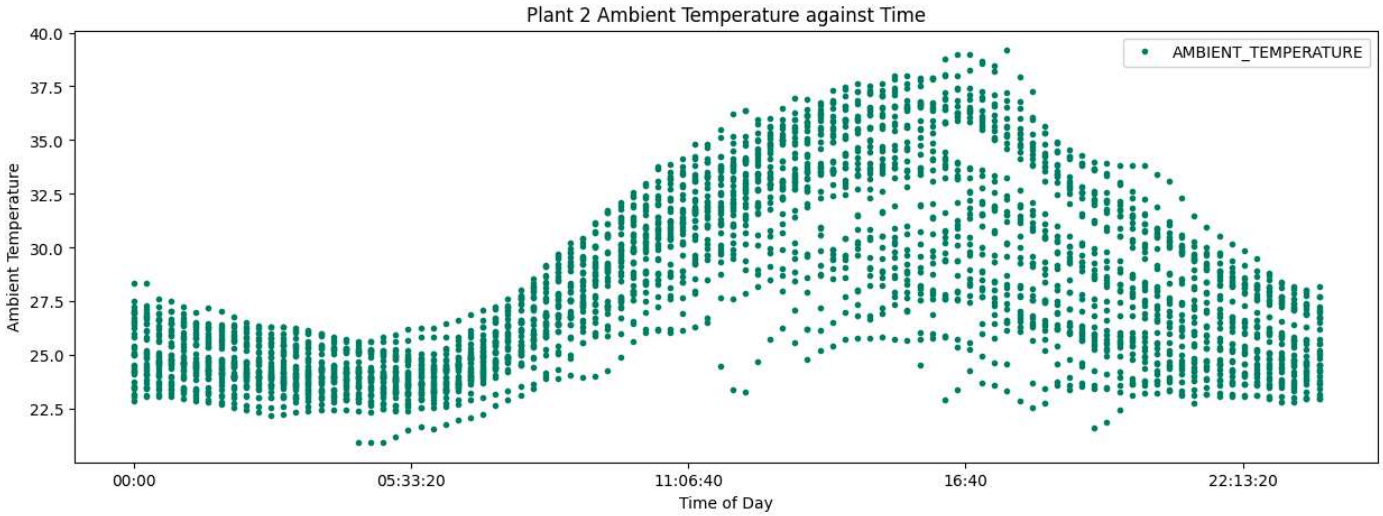

Plant 1 Module Temperature against Time
MODULE_TEMPERATURE
Module Temperature
Time of Day
00:00
05:33:20
11:06:40
16:40
22:13:20


Plant 1 Ambient Temperature against Time
AMBIENT_TEMPERATURE
Ambient Temperature
Time of Day
00:00
05:33:20
11:06:40
16:40
22:13:20

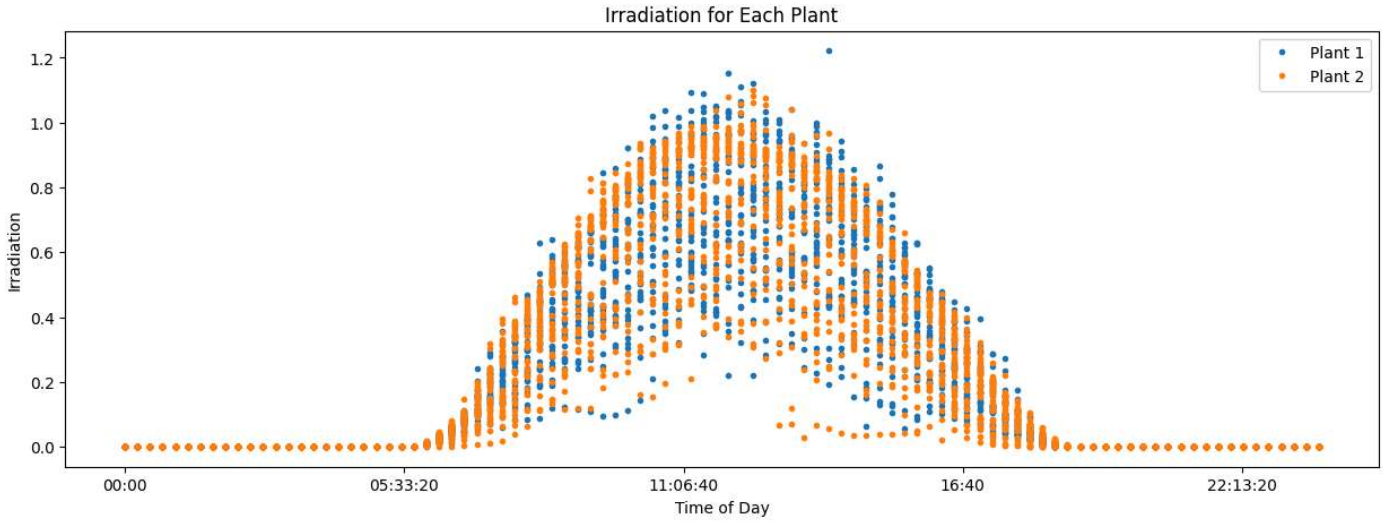

Plant 2 Irradiation against Time
IRRADIATION
Irradiation

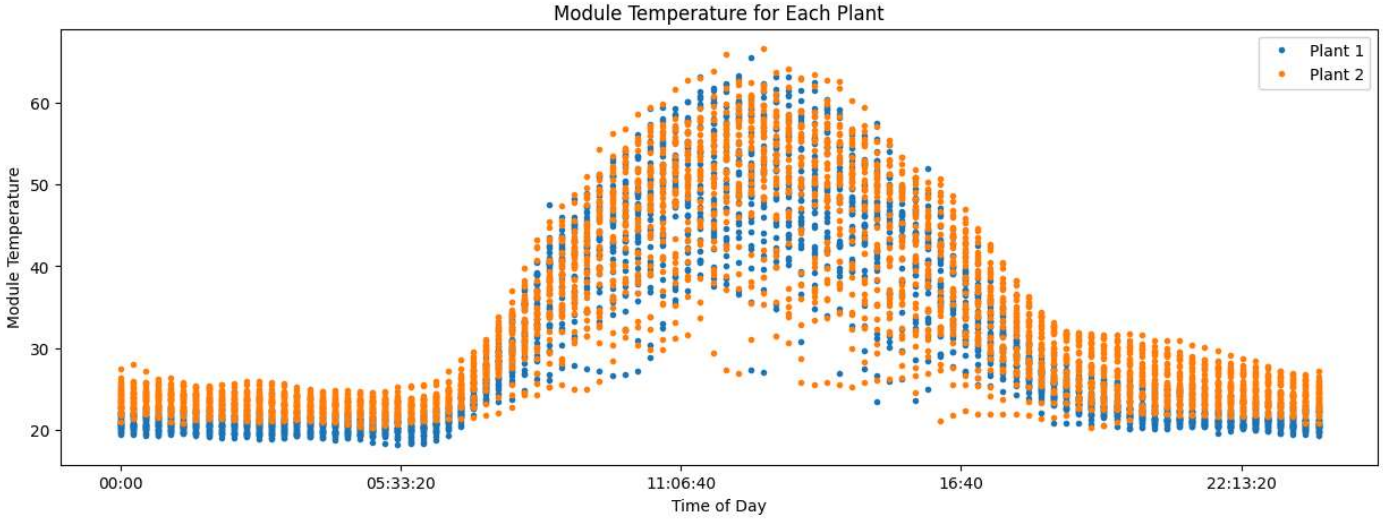

0.2
0.0
00:00
05:33:20
11:06:40
16:40
22:13:20
Time of Day

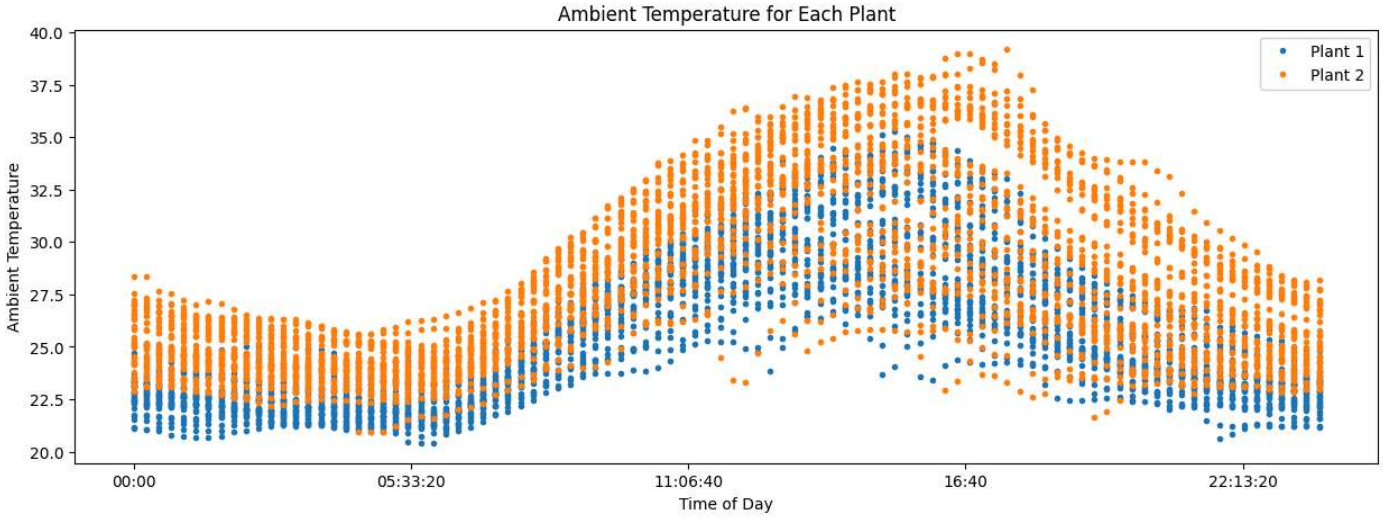

Plant 2 Module Temperature against Time
MODULE_TEMPERATURE
Module Temperature
60
50
40
30
20
00:00
05:33:20
11:06:40
16:40
22:13:20
Time of Day


Plant 2 Ambient Temperature against Time
AMBIENT_TEMPERATURE
Ambient Temperature
40.0
37.5
35.0
32.5
30.0
27.5
25.0
22.5
00:00
05:33:20
11:06:40
16:40
22:13:20
Time of Day

```
# Hourly sens for each plant

# Irradiation
irr_hour = plant1_sensdaily.plot(x= 'TIME', y='IRRADIATION', style='.', figsize = (15, 5),
                                 legend=True, label='Plant 1')
plant2_sensdaily.plot(x='TIME', y='IRRADIATION', style='.', label='Plant 2', ax=irr_hour)
plt.ylabel('Irradiation')
plt.xlabel('Time of Day')
plt.title('Irradiation for Each Plant')
plt.show()

# Module temperature
modtemp_hour = plant1_sensdaily.plot(x= 'TIME', y='MODULE_TEMPERATURE', style='.', figsize = (15, 5),
                                 legend=True, label='Plant 1')
plant2_sensdaily.plot(x='TIME', y='MODULE_TEMPERATURE', style='.', label='Plant 2', ax=modtemp_hour)
plt.ylabel('Module Temperature')
plt.xlabel('Time of Day')
plt.title('Module Temperature for Each Plant')
plt.show()

# Ambient temperature
ambtemp_hour = plant1_sensdaily.plot(x= 'TIME', y='AMBIENT_TEMPERATURE', style='.', figsize = (15, 5),
                                 legend=True, label='Plant 1')
plant2_sensdaily.plot(x='TIME', y='AMBIENT_TEMPERATURE', style='.', label='Plant 2', ax=ambtemp_hour)
plt.ylabel('Ambient Temperature')
plt.xlabel('Time of Day')
plt.title('Ambient Temperature for Each Plant')
plt.show()
```


Irradiation for Each Plant
Plant 1
Plant 2
Irradiation
1.2
1.0
0.8
0.6
0.4
0.2
0.0
00:00
05:33:20
11:06:40
16:40
22:13:20
Time of Day


Module Temperature for Each Plant
Plant 1
Plant 2
Module Temperature
60
50
40
30
20
00:00
05:33:20
11:06:40
16:40
22:13:20
Time of Day


Ambient Temperature for Each Plant
Plant 1
Plant 2
Ambient Temperature
40.0
37.5
35.0
32.5
30.0
27.5
25.0
22.5
20.0
00:00
05:33:20
11:06:40
16:40
22:13:20
Time of Day

```
mergedata1 = plant1_sensdaily.merge(plant1_gendaily, left_on='DATE_TIME', right_on='DATE_TIME')
mergedata1.head()
```

	DATE_TIME	AMBIENT_TEMPERATURE	MODULE_TEMPERATURE	IRRADIATION	SOURCE_KEY	TIME_x	DATE_x	DC_POWER	AC_POWER	DAILY_YIELD
0	2020-05-15 00:00:00	25.184316	22.857507	0.0	HmiyD2TTLFNqkNe	00:00:00	2020-05-15	0.0	0.0	0.0
1	2020-05-15 00:15:00	25.084589	22.761668	0.0	HmiyD2TTLFNqkNe	00:15:00	2020-05-15	0.0	0.0	0.0
2	2020-05-15	24.935753	22.592306	0.0	HmiyD2TTLFNqkNe	00:30:00	2020-	0.0	0.0	0.0

```
mergedata2 = plant2_sensdaily.merge(plant1_gendaily, left_on='DATE_TIME', right_on='DATE_TIME')
mergedata2.head()
```

	DATE_TIME	AMBIENT_TEMPERATURE	MODULE_TEMPERATURE	IRRADIATION	SOURCE_KEY	TIME_x	DATE_x	DC_POWER	AC_POWER	DAILY_YIELD
0	2020-05-15 00:00:00	27.004764	25.060789	0.0	iq8k7ZNt4Mwm3w0	00:00:00	2020-05-15	0.0	0.0	0.0
1	2020-05-15 00:15:00	26.880811	24.421869	0.0	iq8k7ZNt4Mwm3w0	00:15:00	2020-05-15	0.0	0.0	0.0
2	2020-05-15	26.682055	24.427290	0.0	iq8k7ZNt4Mwm3w0	00:30:00	2020-	0.0	0.0	0.0

```
mergedata1
```

	DATE_TIME	AMBIENT_TEMPERATURE	MODULE_TEMPERATURE	IRRADIATION	SOURCE_KEY	DC_POWER	AC_POWER	DAILY_YIELD	TOTAL_YIELD
0	2020-05-15 00:00:00	25.184316	22.857507	0.0	HmiyD2TTLFNqkNe	0.0	0.0	0.000000	143581676.0
1	2020-05-15 00:15:00	25.084589	22.761668	0.0	HmiyD2TTLFNqkNe	0.0	0.0	0.000000	143581676.0
2	2020-05-15 00:30:00	24.935753	22.592306	0.0	HmiyD2TTLFNqkNe	0.0	0.0	0.000000	143581676.0
3	2020-05-15 00:45:00	24.846130	22.360852	0.0	HmiyD2TTLFNqkNe	0.0	0.0	0.000000	143581676.0
4	2020-05-15 01:00:00	24.621525	22.165423	0.0	HmiyD2TTLFNqkNe	0.0	0.0	0.000000	150761642.0
...	...	...	...	...	...	...	...	...	...
	2020-06-								

```
mergedata2
```

	DATE_TIME	AMBIENT_TEMPERATURE	MODULE_TEMPERATURE	IRRADIATION	SOURCE_KEY	DC_POWER	AC_POWER	DAILY_YIELD	TOTAL_YIELD
0	2020-05-15 00:00:00	27.004764	25.060789	0.0	iq8k7ZNt4Mwm3w0	0.0	0.0	0.000000	143581676.0
1	2020-05-15 00:15:00	26.880811	24.421869	0.0	iq8k7ZNt4Mwm3w0	0.0	0.0	0.000000	143581676.0
2	2020-05-15 00:30:00	26.682055	24.427290	0.0	iq8k7ZNt4Mwm3w0	0.0	0.0	0.000000	143581676.0
3	2020-05-15 00:45:00	26.500589	24.420678	0.0	iq8k7ZNt4Mwm3w0	0.0	0.0	0.000000	143581676.0
4	2020-05-15 01:00:00	26.596148	25.088210	0.0	iq8k7ZNt4Mwm3w0	0.0	0.0	0.000000	150761642.0
...	...	...	...	...	...	...	...	...	...
3149	2020-06-17 22:45:00	23.511703	22.856201	0.0	iq8k7ZNt4Mwm3w0	0.0	0.0	129571.000000	156142755.0
3150	2020-06-17 23:00:00	23.482282	22.744190	0.0	iq8k7ZNt4Mwm3w0	0.0	0.0	129571.000000	156142755.0

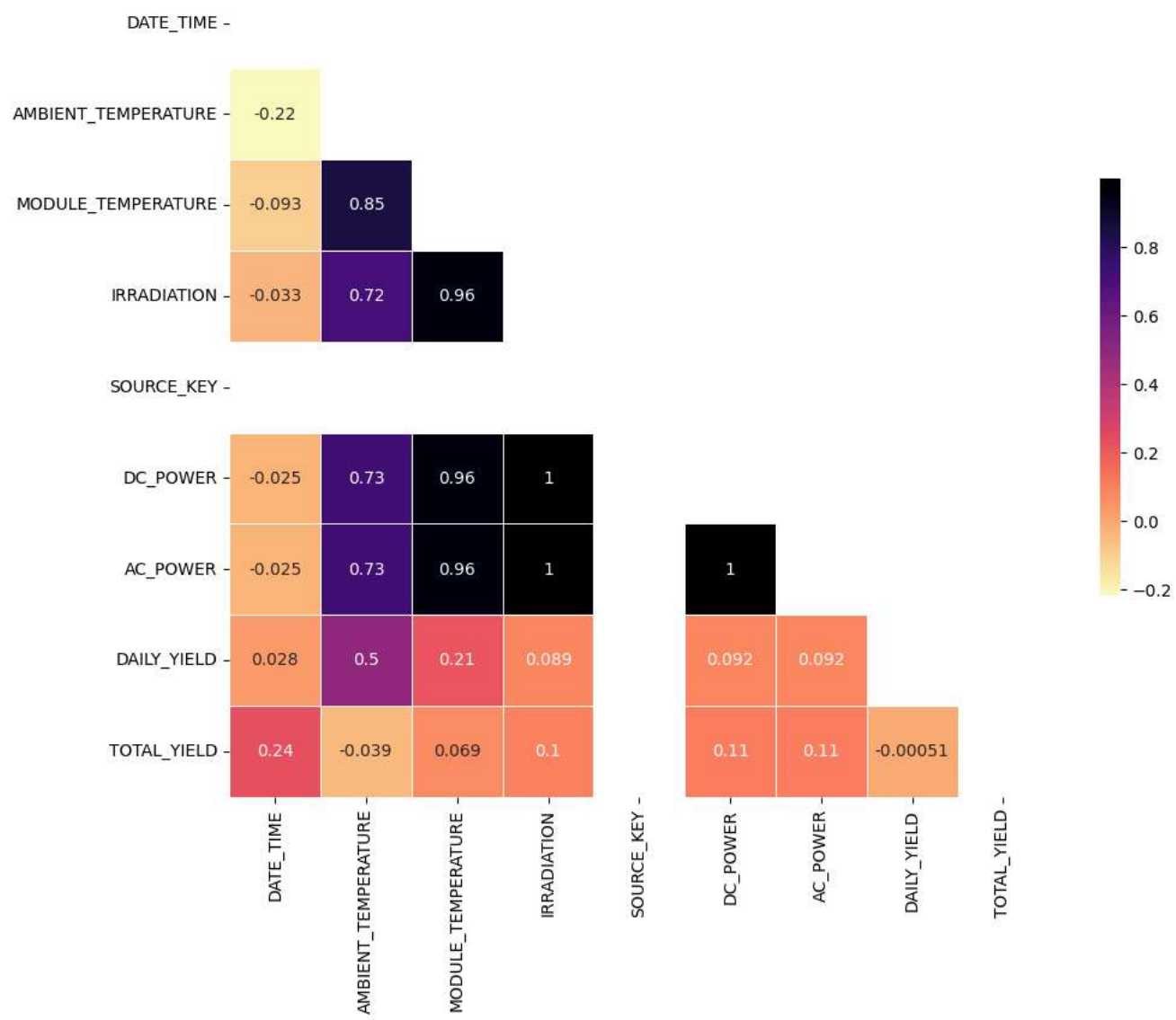
```
# Drop non-numeric columns from the dataset before calculating correlation
numeric_data1 = mergedata1.apply(pd.to_numeric, errors='coerce')  # Convert non-numeric to NaN
numeric_data2 = mergedata2.apply(pd.to_numeric, errors='coerce')

mergecorr1 = numeric_data1.corr()
mergecorr2 = numeric_data2.corr()

print('Plant 1 Merged Generation and Sensor Data Correlation Coefficient')
print(mergecorr1)

print('Plant 2 Merged Generation and Sensor Data Correlation Coefficient')
print(mergecorr2)
```

```
Plant 1 Merged Generation and Sensor Data Correlation Coefficient
                     DATE_TIME  AMBIENT_TEMPERATURE  MODULE_TEMPERATURE  \
DATE_TIME             1.000000            -0.215680           -0.093266
AMBIENT_TEMPERATURE  -0.215680             1.000000            0.853162
MODULE_TEMPERATURE   -0.093266             0.853162            1.000000
IRRADIATION          -0.032842             0.721839            0.961422
SOURCE_KEY                 NaN                  NaN                 NaN
DC_POWER             -0.025361             0.725679            0.960939
AC_POWER             -0.025288             0.725879            0.961011
DAILY_YIELD           0.028455             0.498010            0.212765
TOTAL_YIELD           0.235218            -0.039269            0.069338

                     IRRADIATION  SOURCE_KEY  DC_POWER  AC_POWER  DAILY_YIELD  \
DATE_TIME              -0.032842         NaN -0.025361 -0.025288     0.028455
AMBIENT_TEMPERATURE     0.721839         NaN  0.725679  0.725879     0.498010
MODULE_TEMPERATURE      0.961422         NaN  0.960939  0.961011     0.212765
IRRADIATION             1.000000         NaN  0.995957  0.995864     0.089470
SOURCE_KEY                   NaN         NaN       NaN       NaN          NaN
DC_POWER                0.995957         NaN  1.000000  0.999997     0.092450
AC_POWER                0.995864         NaN  0.999997  1.000000     0.092403
DAILY_YIELD             0.089470         NaN  0.092450  0.092403     1.000000
TOTAL_YIELD             0.102223         NaN  0.106301  0.106401    -0.000510

                     TOTAL_YIELD
DATE_TIME               0.235218
AMBIENT_TEMPERATURE    -0.039269
MODULE_TEMPERATURE      0.069338
IRRADIATION             0.102223
SOURCE_KEY                   NaN
DC_POWER                0.106301
AC_POWER                0.106401
DAILY_YIELD            -0.000510
TOTAL_YIELD             1.000000
Plant 2 Merged Generation and Sensor Data Correlation Coefficient
                     DATE_TIME  AMBIENT_TEMPERATURE  MODULE_TEMPERATURE  \
DATE_TIME             1.000000            -0.308113           -0.181375
AMBIENT_TEMPERATURE  -0.308113             1.000000            0.844636
MODULE_TEMPERATURE   -0.181375             0.844636            1.000000
IRRADIATION          -0.107898             0.663040            0.946294
SOURCE_KEY                 NaN                  NaN                 NaN
DC_POWER             -0.024945             0.613562            0.871068
AC_POWER             -0.024871             0.613717            0.871227
DAILY_YIELD           0.028999             0.511104            0.247288
TOTAL_YIELD           0.235269            -0.108486            0.032117
```

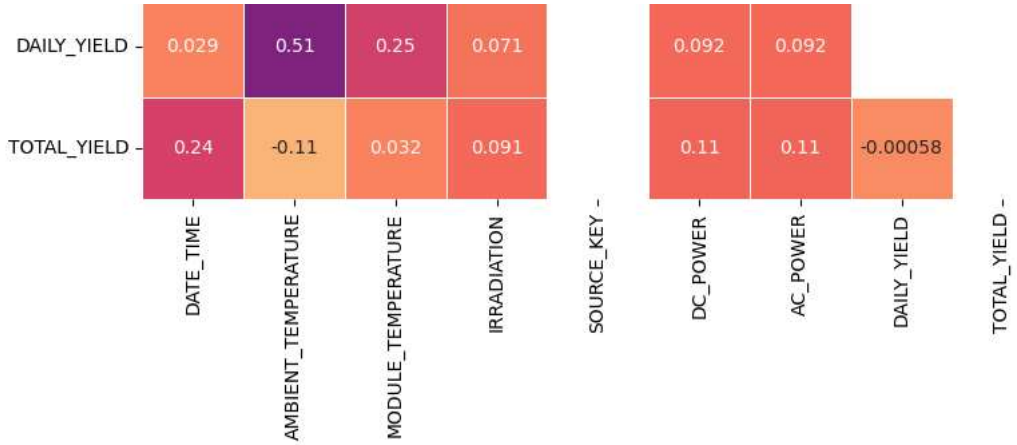
```
                         IRRADIATION  SOURCE_KEY  DC_POWER  AC_POWER  DAILY_YIELD  \
DATE_TIME                  -0.107898         NaN -0.024945 -0.024871     0.028999
AMBIENT_TEMPERATURE         0.663040         NaN  0.613562  0.613717     0.511104
MODULE_TEMPERATURE          0.946294         NaN  0.871068  0.871227     0.247288
IRRADIATION                 1.000000         NaN  0.907776  0.907958     0.071228
SOURCE_KEY                       NaN         NaN       NaN       NaN          NaN
DC_POWER                    0.907776         NaN  1.000000  0.999997     0.092018
AC_POWER                    0.907958         NaN  0.999997  1.000000     0.091970
DAILY_YIELD                 0.071228         NaN  0.092018  0.091970     1.000000
TOTAL_YIELD                 0.090614         NaN  0.106345  0.106446    -0.000585

                         TOTAL_YIELD
```

```
s1mask = np.triu(np.ones_like(mergecorr1, dtype=bool))
s2mask = np.triu(np.ones_like(mergecorr2, dtype=bool))

cmap = 'magma_r'

print('Plant 1')
f, ax = plt.subplots(figsize=(11, 9))
sbn.heatmap(mergecorr1, mask=s1mask, cmap=cmap, square=True, linewidths=.5, cbar_kws={"shrink": .5}, annot=True)
plt.show()

print('Plant 2')
f, ax = plt.subplots(figsize=(11, 9))
sbn.heatmap(mergecorr2, mask=s2mask, cmap=cmap, square=True, linewidths=.5, cbar_kws={"shrink": .5}, annot=True)
plt.show()
```

Plant 1



Plant 2



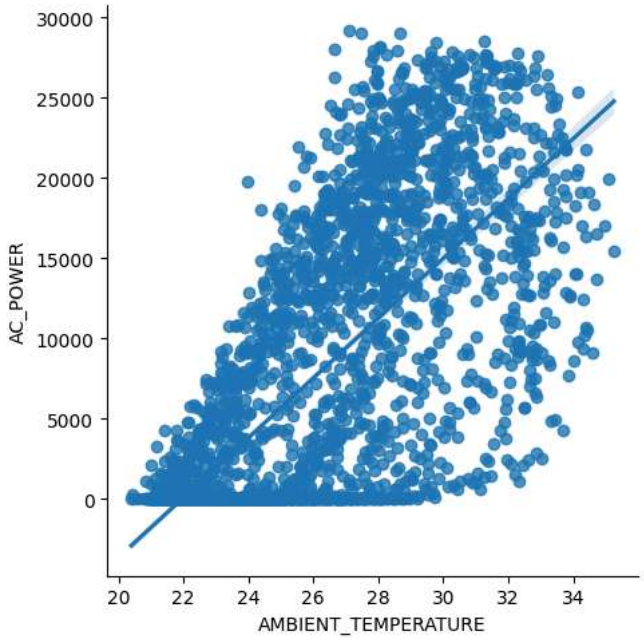

```
# Plant 1
c1 = mergecorr1.unstack()
sort1 = c1.sort_values(kind="quicksort")
print('Plant 1 Top Correlations:')
print(sort1[22:42])
print('')

# Plant 2
c2 = mergecorr2.unstack()
sort2 = c2.sort_values(kind="quicksort")
print('Plant 2 Top Correlations:')
print(sort2[22:42])
```

```
Plant 1 Top Correlations:
DC_POWER             DAILY_YIELD            0.092450
DAILY_YIELD          DC_POWER               0.092450
IRRADIATION          TOTAL_YIELD            0.102223
TOTAL_YIELD          IRRADIATION            0.102223
DC_POWER             TOTAL_YIELD            0.106301
TOTAL_YIELD          DC_POWER               0.106301
AC_POWER             TOTAL_YIELD            0.106401
TOTAL_YIELD          AC_POWER               0.106401
DAILY_YIELD          MODULE_TEMPERATURE     0.212765
MODULE_TEMPERATURE   DAILY_YIELD            0.212765
TOTAL_YIELD          DATE_TIME              0.235218
DATE_TIME            TOTAL_YIELD            0.235218
AMBIENT_TEMPERATURE  DAILY_YIELD            0.498010
DAILY_YIELD          AMBIENT_TEMPERATURE    0.498010
AMBIENT_TEMPERATURE  IRRADIATION            0.721839
IRRADIATION          AMBIENT_TEMPERATURE    0.721839
DC_POWER             AMBIENT_TEMPERATURE    0.725679
AMBIENT_TEMPERATURE  DC_POWER               0.725679
                     AC_POWER               0.725879
AC_POWER             AMBIENT_TEMPERATURE    0.725879
dtype: float64

Plant 2 Top Correlations:
DAILY_YIELD          AC_POWER               0.091970
AC_POWER             DAILY_YIELD            0.091970
DC_POWER             DAILY_YIELD            0.092018
DAILY_YIELD          DC_POWER               0.092018
DC_POWER             TOTAL_YIELD            0.106345
TOTAL_YIELD          DC_POWER               0.106345
AC_POWER             TOTAL_YIELD            0.106446
TOTAL_YIELD          AC_POWER               0.106446
                     DATE_TIME              0.235269
DATE_TIME            TOTAL_YIELD            0.235269
DAILY_YIELD          MODULE_TEMPERATURE     0.247288
MODULE_TEMPERATURE   DAILY_YIELD            0.247288
AMBIENT_TEMPERATURE  DAILY_YIELD            0.511104
DAILY_YIELD          AMBIENT_TEMPERATURE    0.511104
DC_POWER             AMBIENT_TEMPERATURE    0.613562
AMBIENT_TEMPERATURE  DC_POWER               0.613562
                     AC_POWER               0.613717
AC_POWER             AMBIENT_TEMPERATURE    0.613717
AMBIENT_TEMPERATURE  IRRADIATION            0.663040
IRRADIATION          AMBIENT_TEMPERATURE    0.663040
dtype: float64
```

```
sbn.lmplot(x='AMBIENT_TEMPERATURE', y='AC_POWER', data=mergedata1)
print('Plant 1')
```

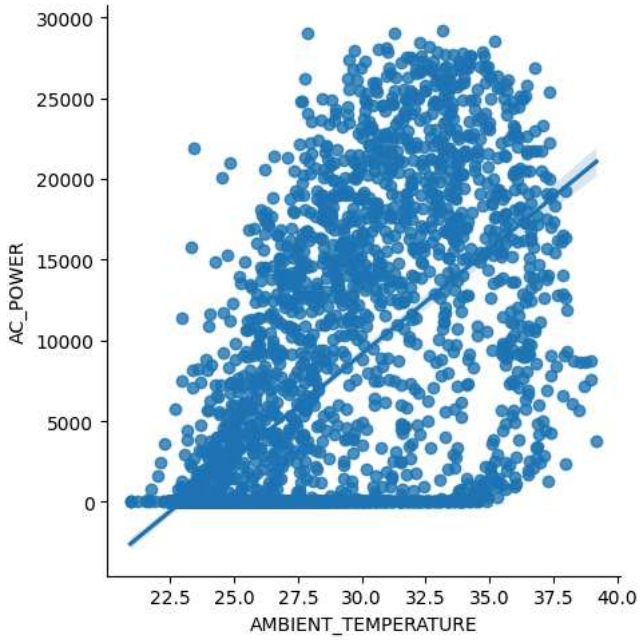
```
plt.show()
```

```
Plant 1
```




```
sbn.lmplot(x='AMBIENT_TEMPERATURE', y='AC_POWER', data=mergedata2)
print('Plant 2')
plt.show()
```

```
Plant 2
```




```
mergedata1.columns
```

```
Index(['DATE_TIME', 'AMBIENT_TEMPERATURE', 'MODULE_TEMPERATURE', 'IRRADIATION',
       'DC_POWER', 'AC_POWER', 'DAILY_YIELD', 'TOTAL_YIELD'],
      dtype='object')
```

```
mergedata2.columns
```

```
Index(['DATE_TIME', 'AMBIENT_TEMPERATURE', 'MODULE_TEMPERATURE', 'IRRADIATION',
       'DC_POWER', 'AC_POWER', 'DAILY_YIELD', 'TOTAL_YIELD'],
      dtype='object')
```

MODEL

```
X = mergedata1[['AMBIENT_TEMPERATURE', 'MODULE_TEMPERATURE', 'IRRADIATION']]
y = mergedata1['AC_POWER']


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)


from sklearn.linear_model import LinearRegression


lm = LinearRegression()


lm.fit(X_train, y_train)
```

```
LinearRegression()
```

```
print('PLANT 1')
print('The intercept for the linear regression is at', lm.intercept_)
print('The linear regression coefficients are:')

coef_df = pd.DataFrame(lm.coef_, X.columns, columns=['Coeff'])
print(coef_df)
```

```
PLANT 1
The intercept for the linear regression is at -440.7989426058075
The linear regression coefficients are:
                         Coeff
AMBIENT_TEMPERATURE     -2.859859
MODULE_TEMPERATURE      30.129244
IRRADIATION          27327.862959
```

```
X2 = mergedata2[['AMBIENT_TEMPERATURE', 'MODULE_TEMPERATURE', 'IRRADIATION']] # Features
y2 = mergedata2['AC_POWER']


X2_train, X2_test, y2_train, y2_test = train_test_split(X2, y2, test_size=0.2)


lm2 = LinearRegression()
lm2.fit(X2_train, y2_train)
```

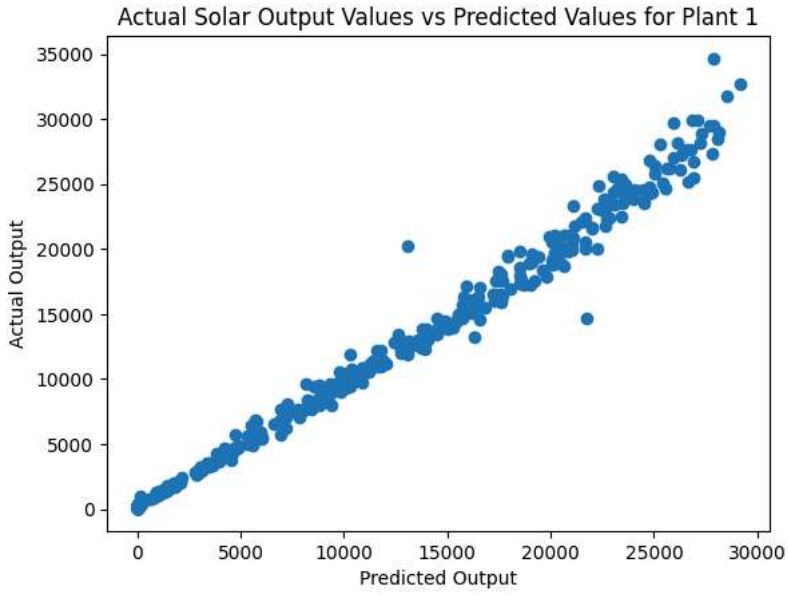
```
LinearRegression()
```

```
print('PLANT 2')
print('The intercept for the linear regression is at', lm2.intercept_)
print('The linear regression coefficients are:')

coef2_df = pd.DataFrame(lm2.coef_, X2.columns, columns=['Coeff2'])
print(coef2_df)
```

```
PLANT 2
The intercept for the linear regression is at 966.4619037322764
The linear regression coefficients are:
                          Coeff2
AMBIENT_TEMPERATURE   -259.465650
MODULE_TEMPERATURE     267.516200
IRRADIATION          18041.434265
```

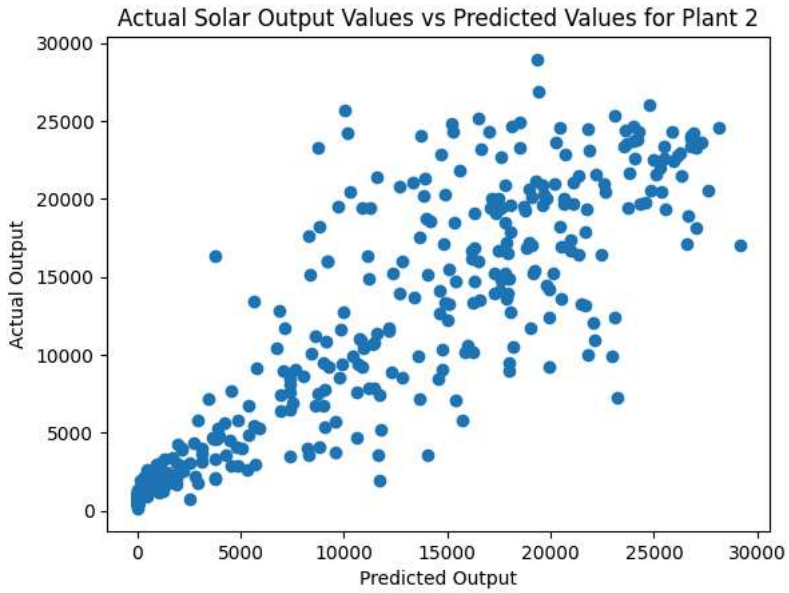
```
predictions1 = lm.predict(X_test)
predictions2 = lm2.predict(X2_test)


plt.scatter(y_test, predictions1)
plt.title('Actual Solar Output Values vs Predicted Values for Plant 1')
plt.xlabel('Predicted Output')
plt.ylabel('Actual Output')
```

```
Text(0, 0.5, 'Actual Output')
```




```
plt.scatter(y2_test, predictions2)
plt.title('Actual Solar Output Values vs Predicted Values for Plant 2')
plt.xlabel('Predicted Output')
plt.ylabel('Actual Output')
```

```
Text(0, 0.5, 'Actual Output')
```

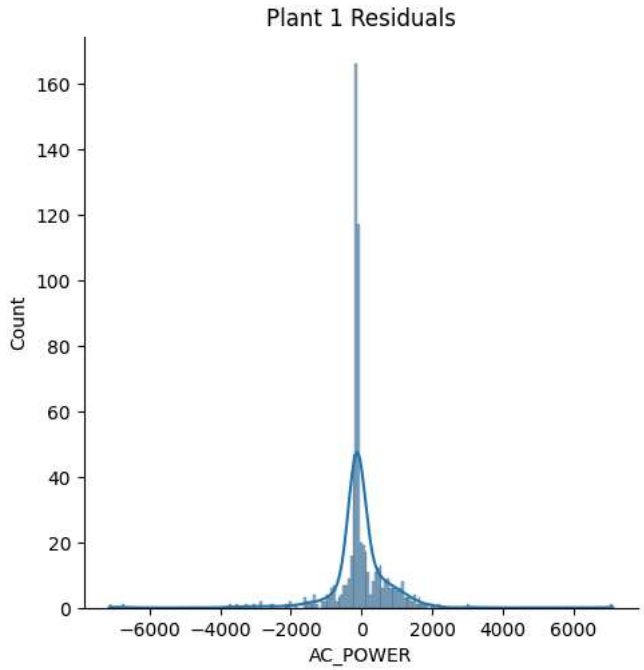



```
from sklearn import metrics

MAE1 = metrics.mean_absolute_error(y_test,predictions1)
MSE1 = metrics.mean_squared_error(y_test,predictions1)
RMSE1 = np.sqrt(metrics.mean_squared_error(y_test,predictions1))
print('Metrics for Plant 1 Linear Model')
print('MAE: ', MAE1)
print('MSE: ',MSE1)
print('RMSE: ', RMSE1)
print()

MAE1 = metrics.mean_absolute_error(y2_test,predictions2)
MSE1 = metrics.mean_squared_error(y2_test,predictions2)
RMSE1 = np.sqrt(metrics.mean_squared_error(y2_test,predictions2))
print('Metrics for Plant 2 Linear Model')
print('MAE: ', MAE1)
print('MSE: ',MSE1)
print('RMSE: ', RMSE1)
```

```
Metrics for Plant 1 Linear Model
MAE:  445.19195746063843
MSE:  684795.626970501
RMSE:  827.5237923893797

Metrics for Plant 2 Linear Model
MAE:  2107.6113019156996
MSE:  11317869.920204366
RMSE:  3364.204203107232
```

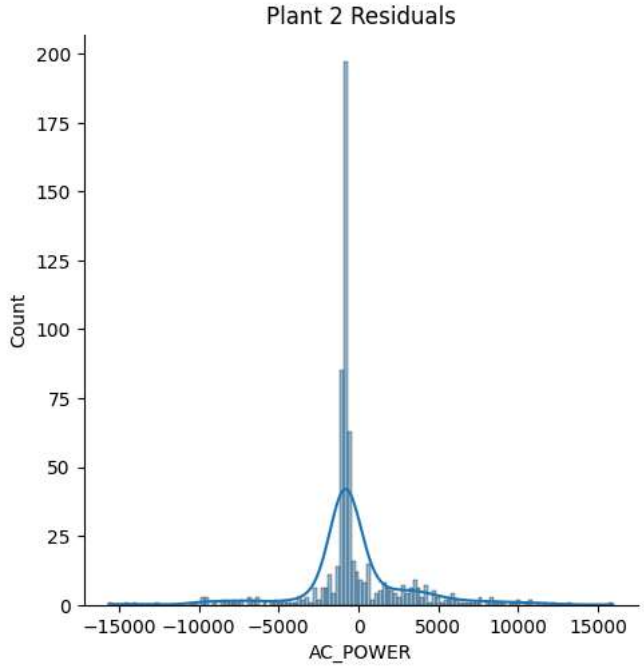
```
sbn.displot((y_test-predictions1), kde=True)
plt.title('Plant 1 Residuals')
```

```
Text(0.5, 1.0, 'Plant 1 Residuals')
```




```
sbn.displot((y2_test-predictions2), kde=True)
plt.title('Plant 2 Residuals')
```

```
Text(0.5, 1.0, 'Plant 2 Residuals')
```



rgb