Exploratory Data Analysis

In [1]:

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

Load Data

In [2]:

```
cwd = os.getcwd()
path_to_challenge1 = os.path.dirname(os.path.dirname(cwd))
path_to_all_data = os.path.join(path_to_challenge1, 'data', 'dataset_500.csv')
path_to_test_data = os.path.join(path_to_challenge1, 'data', 'test_dataset_500.csv')
path_to_train_data = os.path.join(path_to_challenge1, 'data', 'training_dataset_500.
```

In [3]:

```
# All Data
df = pd.read_csv(path_to_all_data)
df_test = pd.read_csv(path_to_test_data)
df_train = pd.read_csv(path_to_train_data)
```

In [4]:

```
df.head()
```

Out[4]:

	ID	Label	House	Year	Month	Temperature	Daylight	EnergyProduction
0	0	0	1	2011	7	26.2	178.9	740
1	1	1	1	2011	8	25.8	169.7	731
2	2	2	1	2011	9	22.8	170.2	694
3	3	3	1	2011	10	16.4	169.1	688
4	4	4	1	2011	11	11.4	169.1	650

In [5]:

```
df_test.head()
```

Out[5]:

	ID	Label	House	Year	Month	Temperature	Daylight	EnergyProduction
0	23	23	1	2013	6	22.0	125.5	778
1	47	23	2	2013	6	21.1	123.1	627
2	71	23	3	2013	6	21.9	126.8	735
3	95	23	4	2013	6	20.2	125.2	533
4	119	23	5	2013	6	20.2	125.2	533

In [6]:

```
len(df_test)
```

Out[6]:

```
500
```

Profiles

In [7]:

```
profile = df.profile_report(title='Pandas Profiling Full Data Report')
profile.to_file(output_file="data_profile_full.html")
```

In [8]:

```
test_profile = df_test.profile_report(title='Pandas Profiling Test Data Report')
test_profile.to_file(output_file="data_profile_test.html")
```

In [9]:

```
train_profile = df_train.profile_report(title='Pandas Profiling Train Data Report')
train_profile.to_file(output_file="data_profile_training.html")
```

In [ ]:

```

```

Manual Exploration

In [10]:

```
df['DateTime'] = pd.to_datetime(df.Year.map(str) + df.Month.map(str), format="%Y%m")
df.set_index('DateTime', inplace=True)
```

In [11]:

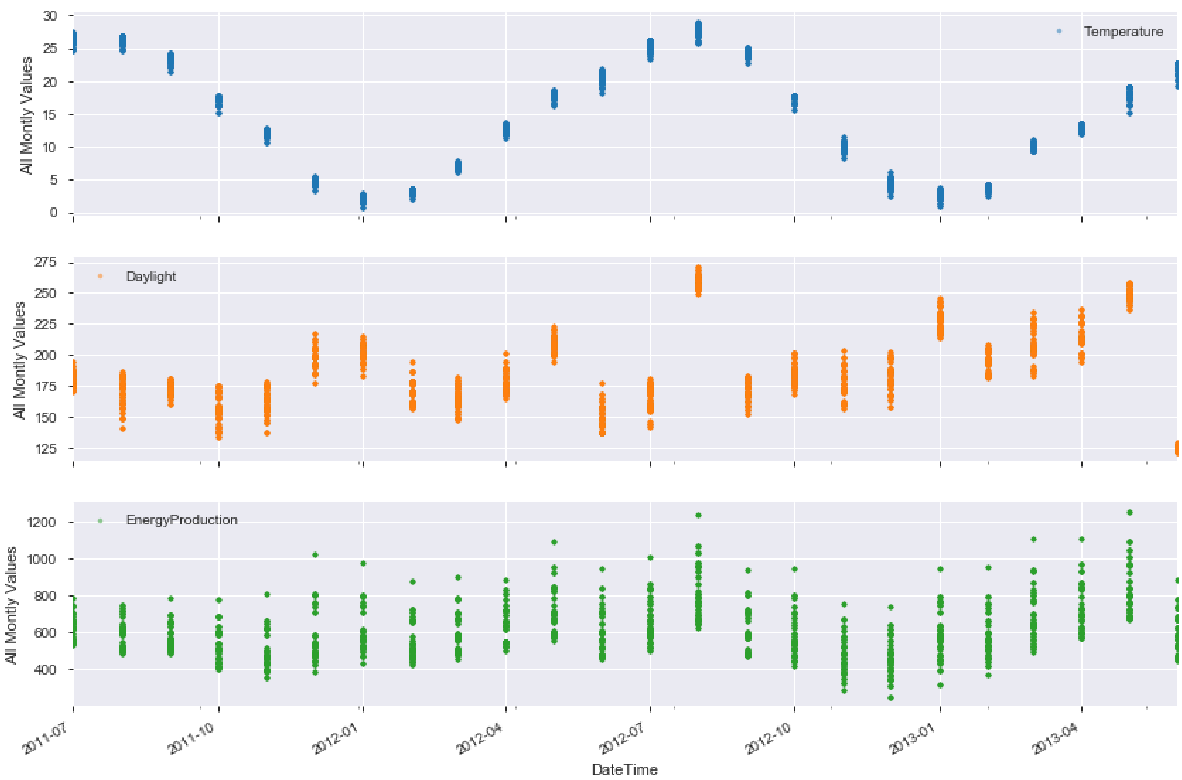
```
df.head()
```

Out[11]:

	ID	Label	House	Year	Month	Temperature	Daylight	EnergyProduction
DateTime								
2011-07-01	0	0	1	2011	7	26.2	178.9	740
2011-08-01	1	1	1	2011	8	25.8	169.7	731
2011-09-01	2	2	1	2011	9	22.8	170.2	694
2011-10-01	3	3	1	2011	10	16.4	169.1	688
2011-11-01	4	4	1	2011	11	11.4	169.1	650

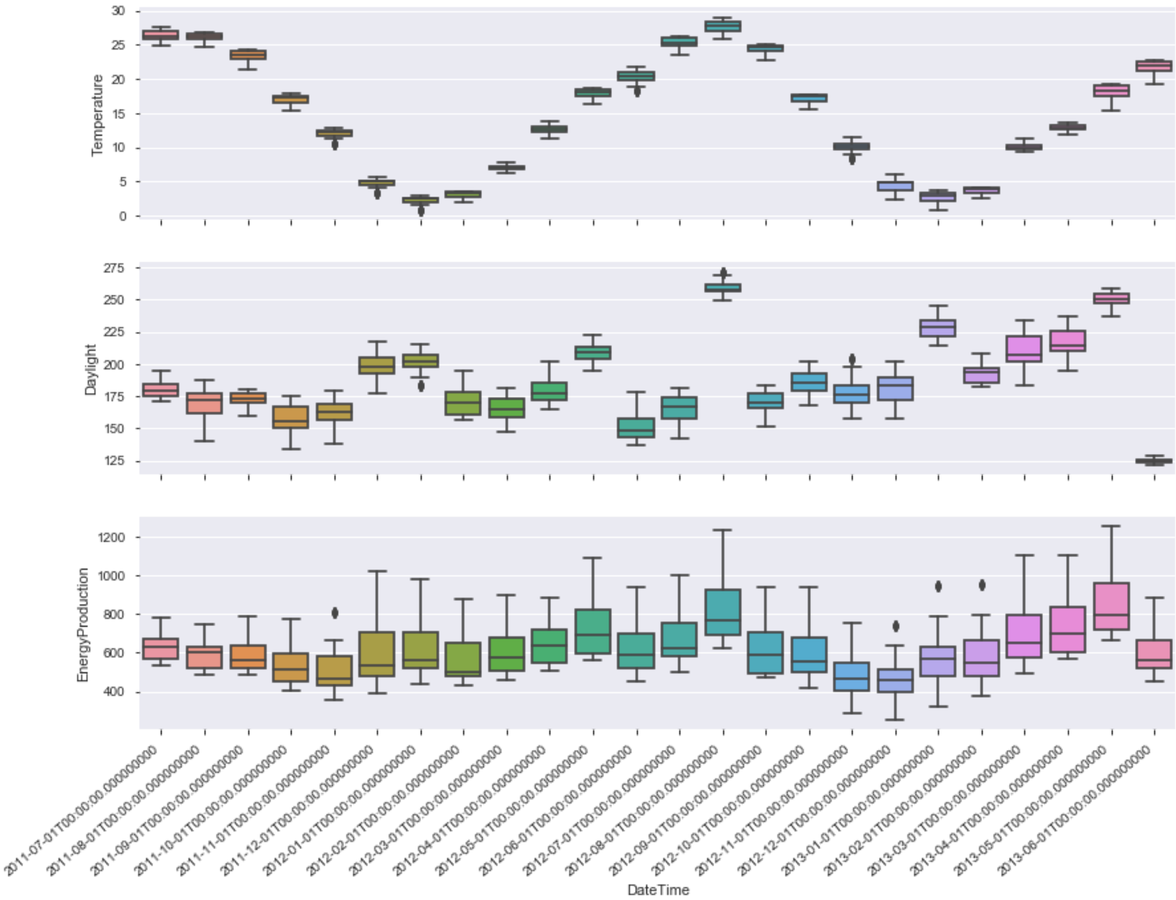
In [12]:

```
# Visualise All values of Timeseries data
cols_plot = ['Temperature', 'Daylight', 'EnergyProduction']
axes = df[cols_plot].plot(marker='.', alpha=0.5, linestyle='None', figsize=(14, 10),
for ax in axes:
    ax.set_ylabel('All Montly Values')
```



In [13]:

```
# Distribution of Time Series
fig, axes = plt.subplots(3, 1, figsize=(14, 10), sharex=True)
for name, ax in zip(cols_plot, axes):
    sns.boxplot(data=df, x=df.index, y=name, ax=ax)
    ax.set_ylabel(name)
    ax.set_xticklabels(ax.get_xticklabels(), rotation=40, ha="right")
    # Remove the automatic x-axis label from all but the bottom subplot
    if ax != axes[-1]:
        ax.set_xlabel('')
```



In [ ]:

```
```

House with Higest and Lowest Values

This plots aim to help visualise how different are houses with large vs low Energy Production

In [14]:

```
group_houses = df.groupby(['House'])[cols_plot].agg(['min', 'max', 'mean', 'std'])
```

In [15]:

```
# Sort DataFrame by "Mean EnergyProduction"
group_houses.sort_values([('EnergyProduction', 'mean')], ascending=False)
```

Out[15]:

	Temperature				Daylight				EnergyProduction		
	min	max	mean	std	min	max	mean	std	min	max	mean
House											
443	2.7	29.0	15.229167	8.862254	129.1	257.5	193.158333	34.706971	698	1254	931.16
395	2.7	29.0	15.229167	8.862254	129.1	257.5	193.158333	34.706971	698	1254	931.16
365	2.7	29.0	15.229167	8.862254	129.1	257.5	193.158333	34.706971	698	1254	931.16
160	2.7	29.0	15.229167	8.862254	129.1	257.5	193.158333	34.706971	698	1254	931.16
109	2.7	29.0	15.229167	8.862254	129.1	257.5	193.158333	34.706971	698	1254	931.16
86	2.7	29.0	15.229167	8.862254	129.1	257.5	193.158333	34.706971	698	1254	931.16
96	2.7	29.0	15.229167	8.862254	129.1	257.5	193.158333	34.706971	698	1254	931.16
277	2.7	29.0	15.229167	8.862254	129.1	257.5	193.158333	34.706971	698	1254	931.16
152	2.7	29.0	15.229167	8.862254	129.1	257.5	193.158333	34.706971	698	1254	931.16
338	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
110	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
447	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
1	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
341	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
81	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
67	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
386	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
281	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
391	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
461	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
15	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
177	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
212	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
190	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
262	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
185	1.8	27.4	14.329167	8.744364	125.5	257.1	189.083333	32.015494	593	1088	799.54
272	2.5	28.4	15.158333	8.836678	126.0	257.9	195.529167	33.161797	637	1046	794.58
348	2.5	28.4	15.158333	8.836678	126.0	257.9	195.529167	33.161797	637	1046	794.58
60	2.5	28.4	15.158333	8.836678	126.0	257.9	195.529167	33.161797	637	1046	794.58
174	2.5	28.4	15.158333	8.836678	126.0	257.9	195.529167	33.161797	637	1046	794.58
...	...	...	...	...	...	...	...	...	...	...	...

	Temperature				Daylight				EnergyProduction		
	min	max	mean	std	min	max	mean	std	min	max	mean
House											
344	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
242	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
346	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
138	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
5	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
92	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
4	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
111	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
406	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
203	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
120	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
292	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
436	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
472	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
410	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
464	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
119	2.7	27.0	14.495833	8.153552	125.2	268.8	180.779167	31.233218	311	723	499.41
263	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
209	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
496	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
441	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
156	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
357	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
469	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
439	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
438	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
275	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
227	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
287	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95
359	2.9	26.2	14.212500	7.839937	125.9	271.3	180.591667	32.593903	357	691	498.95

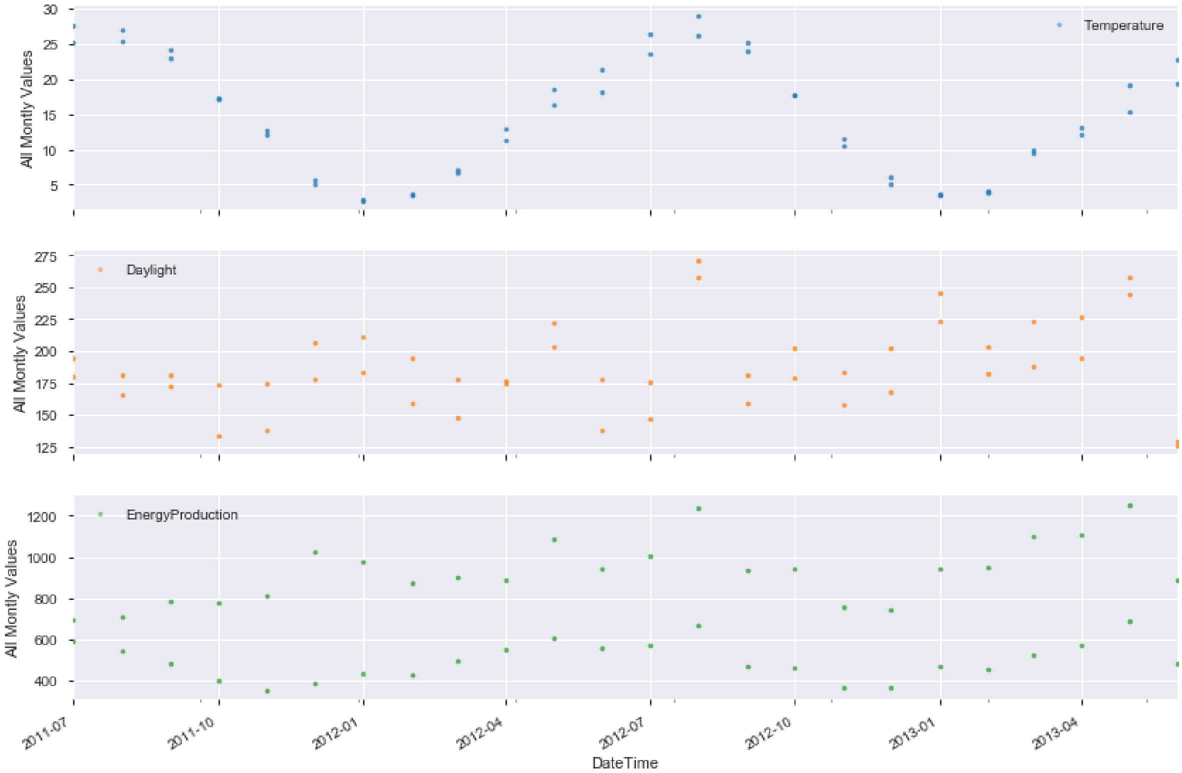
500 rows × 12 columns

In [16]:

```
# Based on the above, lets filter DT
df_filter = df[ df['House'].isin([443,395,287,359]) ]
```

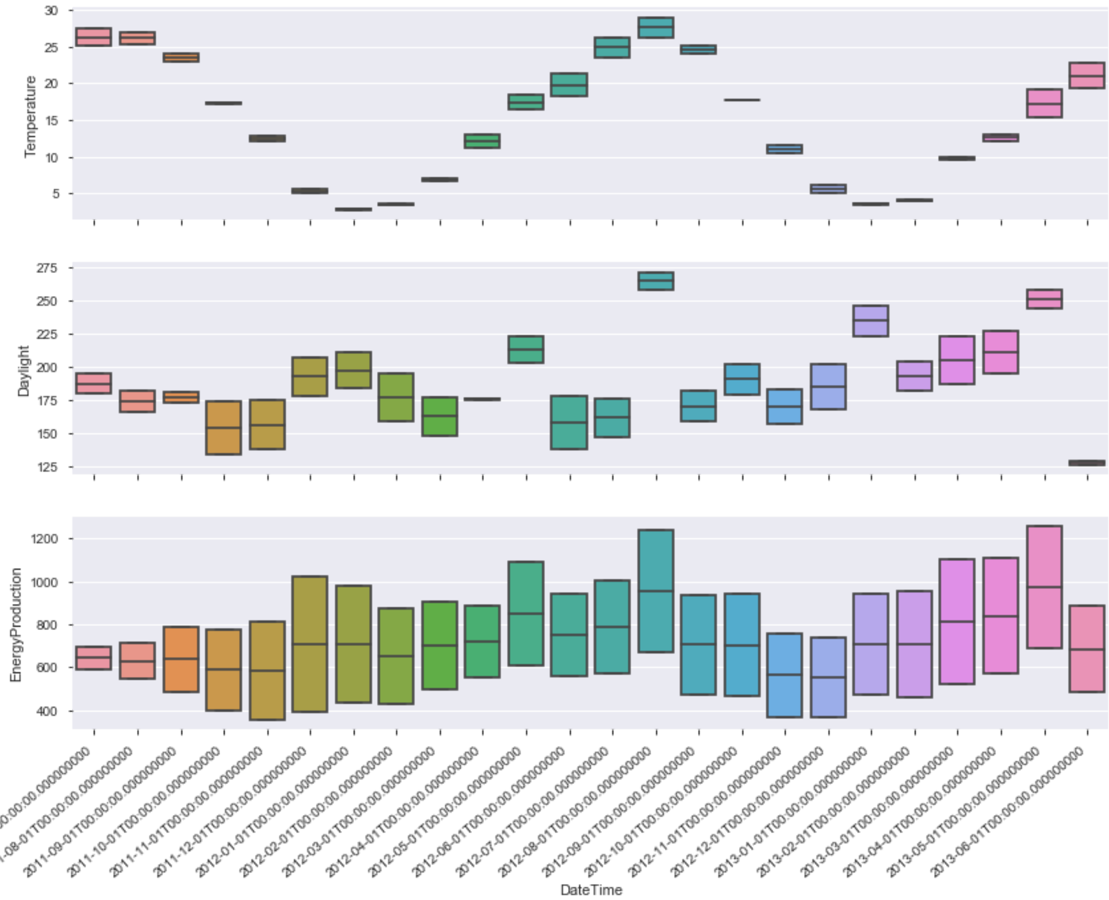
In [17]:

```
axes = df_filter[cols_plot].plot(marker='.', alpha=0.5, linestyle='None', figsize=(1
for ax in axes:
    ax.set_ylabel('All Montly Values')
```



In [18]:

```
# Distribution of Time Series
fig, axes = plt.subplots(3, 1, figsize=(14, 10), sharex=True)
for name, ax in zip(cols_plot, axes):
    sns.boxplot(data=df_filter, x=df_filter.index, y=name, ax=ax)
    ax.set_ylabel(name)
    ax.set_xticklabels(ax.get_xticklabels(), rotation=40, ha="right")
    # Remove the automatic x-axis label from all but the bottom subplot
    if ax != axes[-1]:
        ax.set_xlabel('')
```



This demonstrates that as expected "House" is an Important Feature for any Model, it is also possible to see that there seems to be very high and very low production Houses.

Good to mention that this looks a great dataset in terms of data quality, no missing values, no obvious outliers, etc.

In [ ]:

```

```

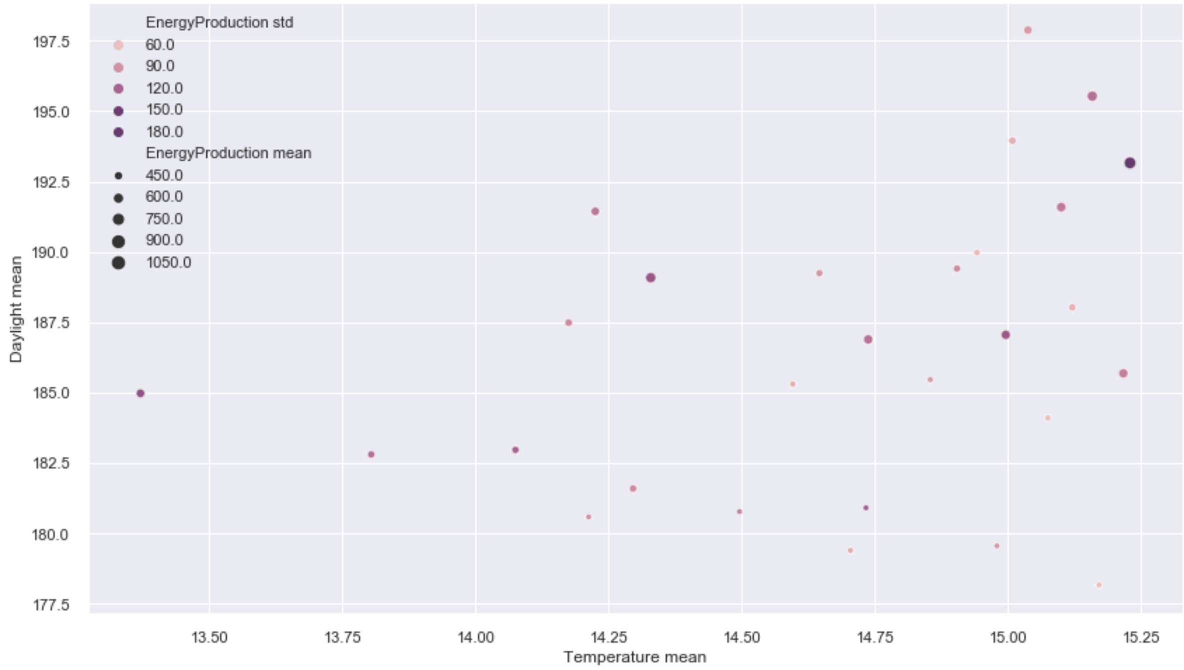
Visualising House Types

In [19]:

```
# Flatening Group House DF
group_houses.columns = [' '.join(col).strip() for col in group_houses.columns.values
```

In [20]:

```
# Plotting Main Variables
sns.set(rc={'figure.figsize':(14,8)})
cmap = sns.cubehelix_palette(dark=.3, light=.8, as_cmap=True)
ax = sns.scatterplot(x="Temperature mean", y="Daylight mean", size="EnergyProductio
                     palette=cmap, alpha=0.5, hue="EnergyProduction std",
                     data=group_houses)
```



In [ ]: