BI Final Project

Group members:

Omer Abid 14922

Syed Maaz Masood 14796

Person working on this document:

Omer Abid 14922

Dataset

Smart Home Dataset with weather Information

https://www.kaggle.com/taranvee/smart-home-dataset-with-weather-information

BI tool

Power BI

Business Knowledge

Top 5/10 business domain info:

- Smart Home Technology comes in two flavors: Central control systems that provides a central interface for controlling devices and App-based systems where different propriety technology is controlled by different apps.
- Currently, controlling your smart devices through hubs are more common. The hubs basically host all your app-based device and then present an integrated interface.
- Connected devices, communicate with each other and us.
- Any electric device can be joined in the network.
- Most devices relate to lightening, home security, entertainment, and thermostat regulation.
- Human can control these devices through interface.
- Internet of things is the central idea to smart homes.
- Boom in Smart phones and tablets computers with their constant internet connection can be configured to control smart home.
- Current systems use radio waves to communicate smart devices include cameras, led lights, motion sensors, door etc.

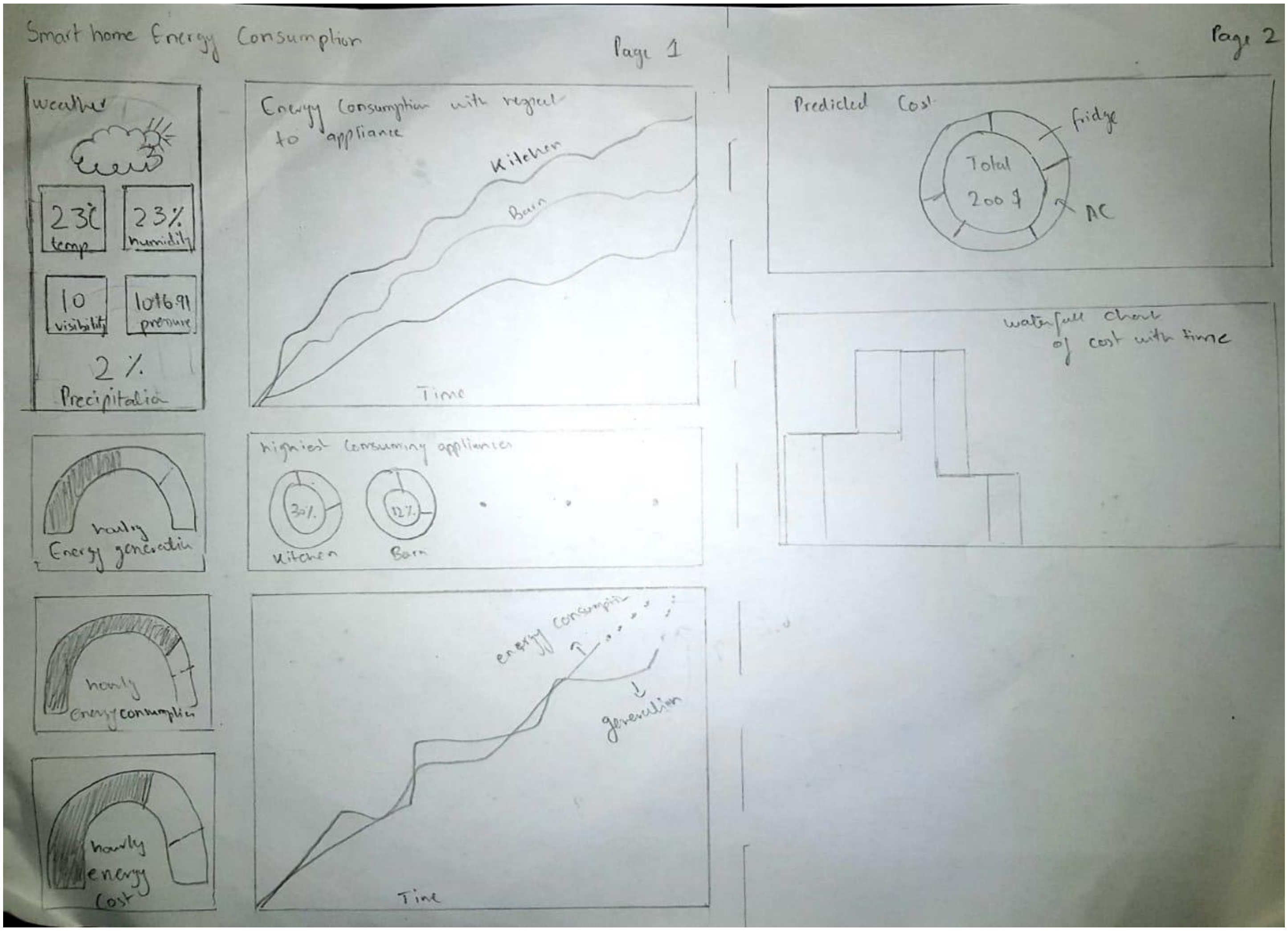
10 related dashboard bulleted list description:

- Different Dashboards on google show smart home analytics focusing on energy usage trend in general using line chart. Then magnifying the analysis using donut chart or displaying figures in numeric for each type of appliance of area.
- Moreover, Another dashboard showed, analysis of cost by using a donut chart to predict cost and showing change in cost using bar charts and then further using a line chart was used to extrapolate energy usage over time.
- Another dashboard along with the details mentioned in point 1 showed weather data in card form.
- In general, many dashboards explored on power consumption with respect to appliances and room using charts like donut and bar charts.

Problem

To analyze the efficiency of smart home energy utilization along with to improve the energy consumption.

Paper story



Story:

On page 1

The story starts by providing user some weather-related information to tell him about the statistics if he wants to analyze the energy generation with respect to it. Then the three gauges basically talk about three important metrics: hourly energy generation, hourly energy consumption and hourly energy cost. Moving on the Energy consumption line chart further add details on how each appliance is consuming the energy and then in relation to this the donut charts highlight the most consuming appliances and shows percent of total consumption. Next, use can see and line graph which plot energy consumption and generation so that user can view how the trend is going and what is the net.

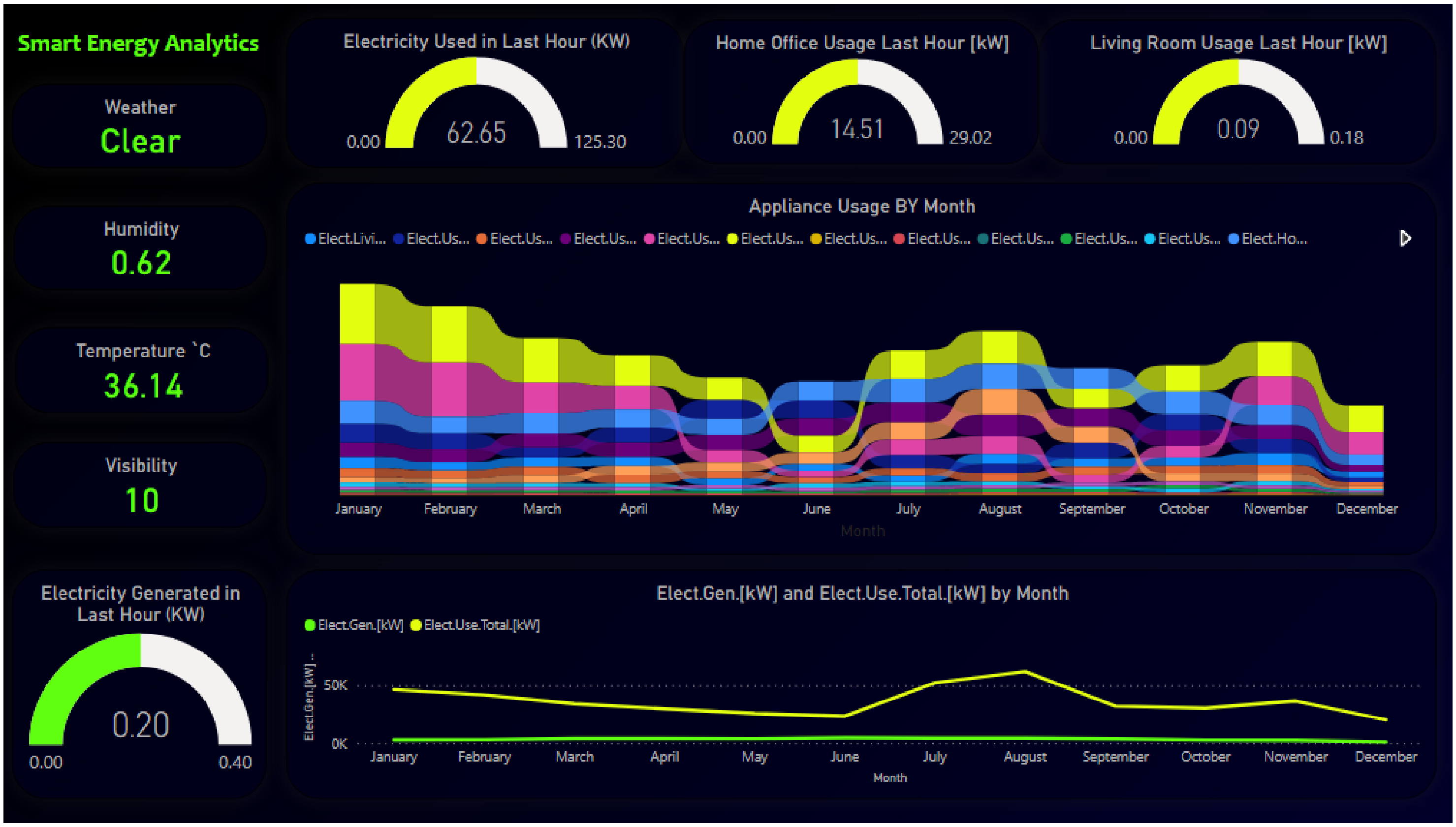
On page 2

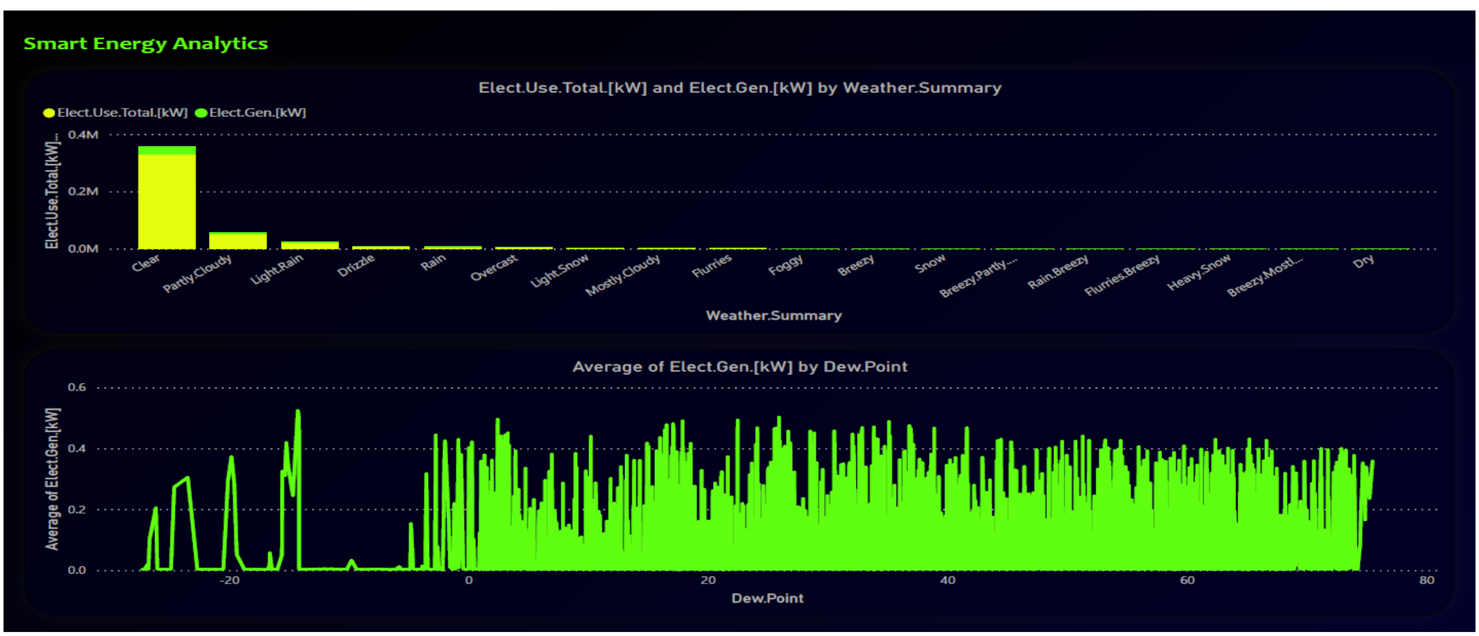
The story shift here more towards the cost side, so the donut chart tells for each appliance expected monthly cost and then lastly, we have a waterfall chart which basically compares the total cost with respect to time.

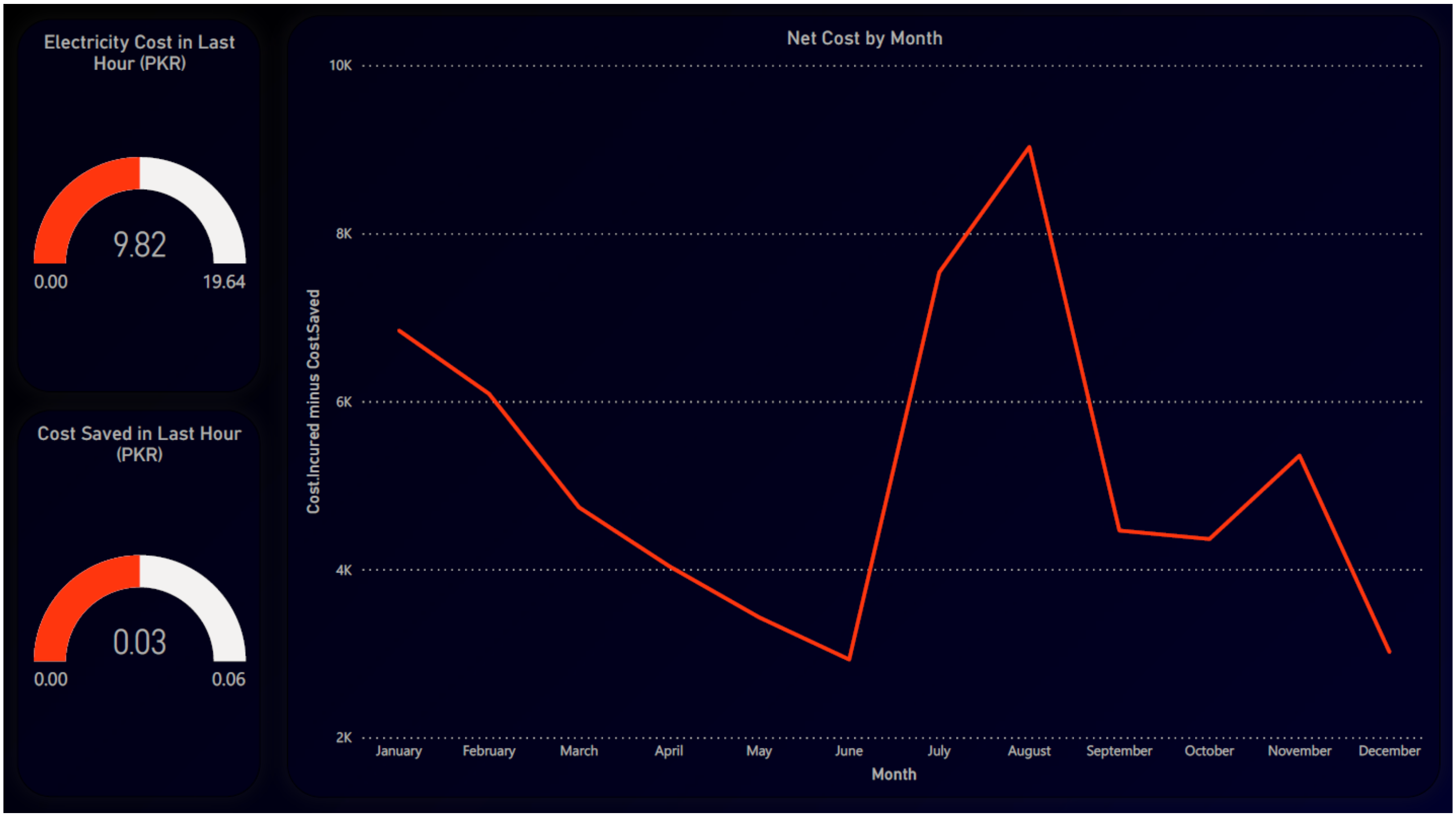
Wrangle

Notebook attached in appendix.

Actual Dashboard







Brief story explanation:

On page 1:

The story starts by giving an overview of current weather perimeter than on the top row we have a series of gauges which gives us last's hour energy stats for overall and certain location which I felt were important. On the left column below the weather stats, we have a gauge showing electricity generated in last hour. These were the operational information. Next, we drill down to see the trend of how the energy is used with respect to months and various appliances from the ribbon chart. Lastly, the line charts present how our energy generation and usage trends with respect to the months. The latter part of this page sort of help to analyze long term trends.

On page 2:

The story continues now to further analyze on energy generation and usage. The first chart continues to tell the how with respect to different weather the energy generation and usage changes and the bottom chart further analyze an energy generation with respect to a weather parameter named dew point. This page continues the analyses of long-term trends which we started in the first page.

On Page 3:

On page three we shift to cost analyses as usual first we see the two gauges on the left to see the ongoing operational statistics about the how much cost we have incurred in the last hour and how much cost we have saved by the energy generation in the last hour. Finally, we shift on long term analysis of the cost, the line chart of net cost with respect to months tells us the overall trend of net cost throughout the year.

Mention group member contribution (5-10 points)

- Maaz helped me with streamlining the overall story that the dashboard shows.
- Guided me over the color scheme on text visibility, specifically.
- Helped In page 2 of the dashboard when making the relation chart of energy generated with respect to dew point.
- Helped in making quick calculated measure of net cost.
- Validated the story both look good and understandable.
- Helped in finding various code for different statistical tests.
- Verified my wrangling activity.
- Helped Selecting dataset.

Link

https://app.powerbi.com/groups/me/reports/dced285b-a86e-4a9f-99e1-c953692dcf29?ctid=fee3b916-01c1-4987-a646-e193432b9eaa

My contribution

How you improved other's (5-10 points)

- Helped selecting color scheme for background and charts.
- Helped in clearing the confusion during wrangling about statistical tests to be done.
- Verified overall story of the BI dashboard.
- Helped selecting dataset.
- Helped on choosing charts in certain section.
- Verified identified problem.

References:

- https://home.howstuffworks.com/smart-home.htm
- https://www.otelco.com/resources/smart-home-guide/

Appendix:

```
In [1]: import pandas as pd
        import numpy as np
        from matplotlib import pyplot
        import time
        from scipy import stats

        from scipy.stats import shapiro
        from scipy.stats import normaltest
        from scipy.stats import anderson
        from scipy.stats import chi2_contingency
        from scipy.stats import chi2

        import statsmodels.api as sm
        from statsmodels.formula.api import ols
        from statsmodels.stats.multicomp import pairwise_tukeyhsd
        from statsmodels.graphics.gofplots import qqplot
```

Import the data

```
In [2]: df = pd.read_csv("HomeC.csv",low_memory=False)
```

Lets have a look at the data

```
In [3]: df.shape
```

```
Out[3]: (503911, 32)
```

```
In [4]: df.head()
```

Out[4]:

	time	use [kW]	gen [kW]	House overall [kW]	Dishwasher [kW]	Furnace 1 [kW]	Furnace 2 [kW]	Home office [kW]	Fridge [kW]	Wine cellar [kW]	...	visibility	summary	apparentTempe
0	1451624400	0.932833	0.003483	0.932833	0.000033	0.020700	0.061917	0.442633	0.124150	0.006983	...	10.0	Clear	
1	1451624401	0.934333	0.003467	0.934333	0.000000	0.020717	0.063817	0.444067	0.124000	0.006983	...	10.0	Clear	

	time	use [kW]	gen [kW]	House overall [kW]	Dishwasher [kW]	Furnace 1 [kW]	Furnace 2 [kW]	Home office [kW]	Fridge [kW]	Wine cellar [kW]	...	visibility	summary	apparentTempe
2	1451624402	0.931817	0.003467	0.931817	0.000017	0.020700	0.062317	0.446067	0.123533	0.006983	...	10.0	Clear	
3	1451624403	1.022050	0.003483	1.022050	0.000017	0.106900	0.068517	0.446583	0.123133	0.006983	...	10.0	Clear	
4	1451624404	1.139400	0.003467	1.139400	0.000133	0.236933	0.063983	0.446533	0.122850	0.006850	...	10.0	Clear	

5 rows × 32 columns

In [5]:
```
df.tail()
```

Out[5]:

	time	use [kW]	gen [kW]	House overall [kW]	Dishwasher [kW]	Furnace 1 [kW]	Furnace 2 [kW]	Home office [kW]	Fridge [kW]	Wine cellar [kW]	...	visibility	summary	apparent
503906	1452128306	1.599333	0.003233	1.599333	0.000050	0.104017	0.625033	0.041750	0.005233	0.008433	...	8.74	Light Rain	
503907	1452128307	1.924267	0.003217	1.924267	0.000033	0.422383	0.637733	0.042033	0.004983	0.008467	...	8.74	Light Rain	
503908	1452128308	1.978200	0.003217	1.978200	0.000050	0.495667	0.620367	0.042100	0.005333	0.008233	...	8.74	Light Rain	
503909	1452128309	1.990950	0.003233	1.990950	0.000050	0.494700	0.634133	0.042100	0.004917	0.008133	...	8.74	Light Rain	
503910	\	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	

5 rows × 32 columns

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
time                  0
use [kW]              1
gen [kW]              1
House overall [kW]    1
Dishwasher [kW]       1
```

```
Furnace 1 [kW]          1
Furnace 2 [kW]          1
Home office [kW]        1
Fridge [kW]             1
Wine cellar [kW]        1
Garage door [kW]        1
Kitchen 12 [kW]         1
Kitchen 14 [kW]         1
Kitchen 38 [kW]         1
Barn [kW]               1
Well [kW]               1
Microwave [kW]          1
Living room [kW]        1
Solar [kW]              1
temperature             1
icon                    1
humidity                1
visibility              1
summary                 1
apparentTemperature     1
pressure                1
windSpeed               1
cloudCover              1
windBearing             1
precipIntensity         1
dewPoint                1
precipProbability       1
dtype: int64
```

From the df.tail and isnull we realize that all the nulls are present in the last row so its logical to delete the nulls

In [7]:
```
df=df.dropna()
```

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
time                    0
use [kW]                0
gen [kW]                0
House overall [kW]      0
Dishwasher [kW]         0
```

```
Furnace 1 [kW]          0
Furnace 2 [kW]          0
Home office [kW]        0
Fridge [kW]             0
Wine cellar [kW]        0
Garage door [kW]        0
Kitchen 12 [kW]         0
Kitchen 14 [kW]         0
Kitchen 38 [kW]         0
Barn [kW]               0
Well [kW]               0
Microwave [kW]          0
Living room [kW]        0
Solar [kW]              0
temperature             0
icon                    0
humidity                0
visibility              0
summary                 0
apparentTemperature     0
pressure                0
windSpeed               0
cloudCover              0
windBearing             0
precipIntensity         0
dewPoint                0
precipProbability       0
dtype: int64
```

In [9]:

```
df.dtypes
```

Out[9]:

```
time                     object
use [kW]                float64
gen [kW]                float64
House overall [kW]      float64
Dishwasher [kW]         float64
Furnace 1 [kW]          float64
Furnace 2 [kW]          float64
Home office [kW]        float64
Fridge [kW]             float64
Wine cellar [kW]        float64
Garage door [kW]        float64
Kitchen 12 [kW]         float64
Kitchen 14 [kW]         float64
Kitchen 38 [kW]         float64
```

```
Barn [kW]                  float64
Well [kW]                  float64
Microwave [kW]             float64
Living room [kW]           float64
Solar [kW]                 float64
temperature                float64
icon                        object
humidity                   float64
visibility                 float64
summary                     object
apparentTemperature        float64
pressure                   float64
windSpeed                  float64
cloudCover                  object
windBearing                float64
precipIntensity            float64
dewPoint                   float64
precipProbability          float64
dtype: object
```

df['time'] = pd.to_datetime(df["time"], unit='m') dont work here since there is some issue with the dataset so lets generate our own date time column with min intervals as stated on the datset website.

In [10]:

```
time = pd.date_range('2016-01-01 05:00', periods=len(df),  freq='min')
time = pd.DatetimeIndex(time)
df['time']=time
```

In [11]:

```
df['time']
```

Out[11]:

```
0         2016-01-01 05:00:00
1         2016-01-01 05:01:00
2         2016-01-01 05:02:00
3         2016-01-01 05:03:00
4         2016-01-01 05:04:00
                  ...
503905    2016-12-16 03:25:00
503906    2016-12-16 03:26:00
503907    2016-12-16 03:27:00
503908    2016-12-16 03:28:00
```

```
503909   2016-12-16 03:29:00
Name: time, Length: 503910, dtype: datetime64[ns]
```

Cloud cover is numeric column but from dataset we can see that it has string values in it in the start, the amount of such values is very less, moreover bfill seems to be best option since such values are in the start and they repeat consecutively.

In [12]:
```
print(df['cloudCover'].value_counts())
#lets replace cloud cover with backfill
df['cloudCover'] = df['cloudCover'].replace('cloudCover', np.nan)
print((df['cloudCover'].isnull().sum()/df.shape[0])*100, '%\n')
df['cloudCover'] = df['cloudCover'].fillna(method='bfill')
df['cloudCover'] = df['cloudCover'].astype(float)
print(df['cloudCover'].value_counts())
```

```
0             68236
0.31          49899
1             48705
0.03          33940
0.04          24117
              ...
0.73            114
0.56             58
0.53             58
cloudCover       58
0.59             57
Name: cloudCover, Length: 78, dtype: int64
0.011509991863626 44 %

0.00    68236
0.31    49899
1.00    48705
0.03    33940
0.04    24117
        ...
0.73      114
0.71      114
0.56       58
0.53       58
```

```
0.59       57
Name: cloudCover, Length: 77, dtype: int64
```

These three columns that are being dropped below have similar columns in the dataset that present the very similar same data.

In [13]:
```
df.drop('use [kW]', axis=1, inplace=True)
df.drop('icon', axis=1, inplace=True)
df.drop('Solar [kW]', axis=1, inplace=True)
```

I want to have cost as a KPI as well, so i have used the price in PKR for KW/min. Cost incured has the price for total use of electricity in house for each minute, Since we are also generating electricity so i have also made a cost.saved column so that we know what is the savings we are making as we generate electricity, that is also in per min.

In [14]:
```
df['Cost.Incured']=0.15676*df['House overall [kW]']
df['Cost.Saved']=0.15676*df['gen [kW]']
```

All the data types now look good.

In [15]:
```
df.dtypes
```

Out[15]:
```
time                      datetime64[ns]
gen [kW]                         float64
House overall [kW]               float64
Dishwasher [kW]                  float64
Furnace 1 [kW]                   float64
Furnace 2 [kW]                   float64
Home office [kW]                 float64
Fridge [kW]                      float64
```

```
Wine cellar [kW]              float64
Garage door [kW]              float64
Kitchen 12 [kW]               float64
Kitchen 14 [kW]               float64
Kitchen 38 [kW]               float64
Barn [kW]                     float64
Well [kW]                     float64
Microwave [kW]                float64
Living room [kW]              float64
temperature                   float64
humidity                      float64
visibility                    float64
summary                        object
apparentTemperature           float64
pressure                      float64
windSpeed                     float64
cloudCover                    float64
windBearing                   float64
precipIntensity               float64
dewPoint                      float64
precipProbability             float64
Cost.Incured                  float64
Cost.Saved                    float64
dtype: object
```

anova

In [16]:

```
y = df['House overall [kW]']
model = ols('y ~ C(Q("summary"))', data=df).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print ("\nAnova => House overall Usage - summary")
display(anova_table)

y = df['gen [kW]']
model = ols('y ~ C(Q("summary"))', data=df).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print ("\nAnova => Energy Generation - summary")
display(anova_table)
```

```
Anova => House overall Usage - summary
```

	sum_sq	df	F	PR(>F)

	sum_sq	df	F	PR(>F)
C(Q("summary"))	1462.918537	17.0	77.044636	8.493588e-268
Residual	562815.796864	503892.0	NaN	NaN

```
Anova => Energy Generation - summary
```

	sum_sq	df	F	PR(>F)
C(Q("summary"))	32.160311	17.0	115.139148	0.0
Residual	8279.150244	503892.0	NaN	NaN

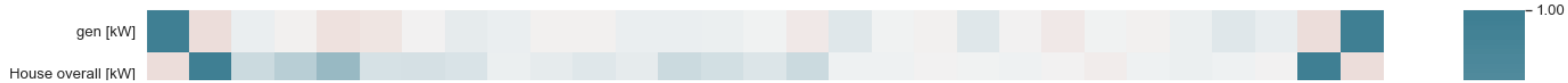
Anova result shows that overall use of electricity and Electric generation against weather summary is significant

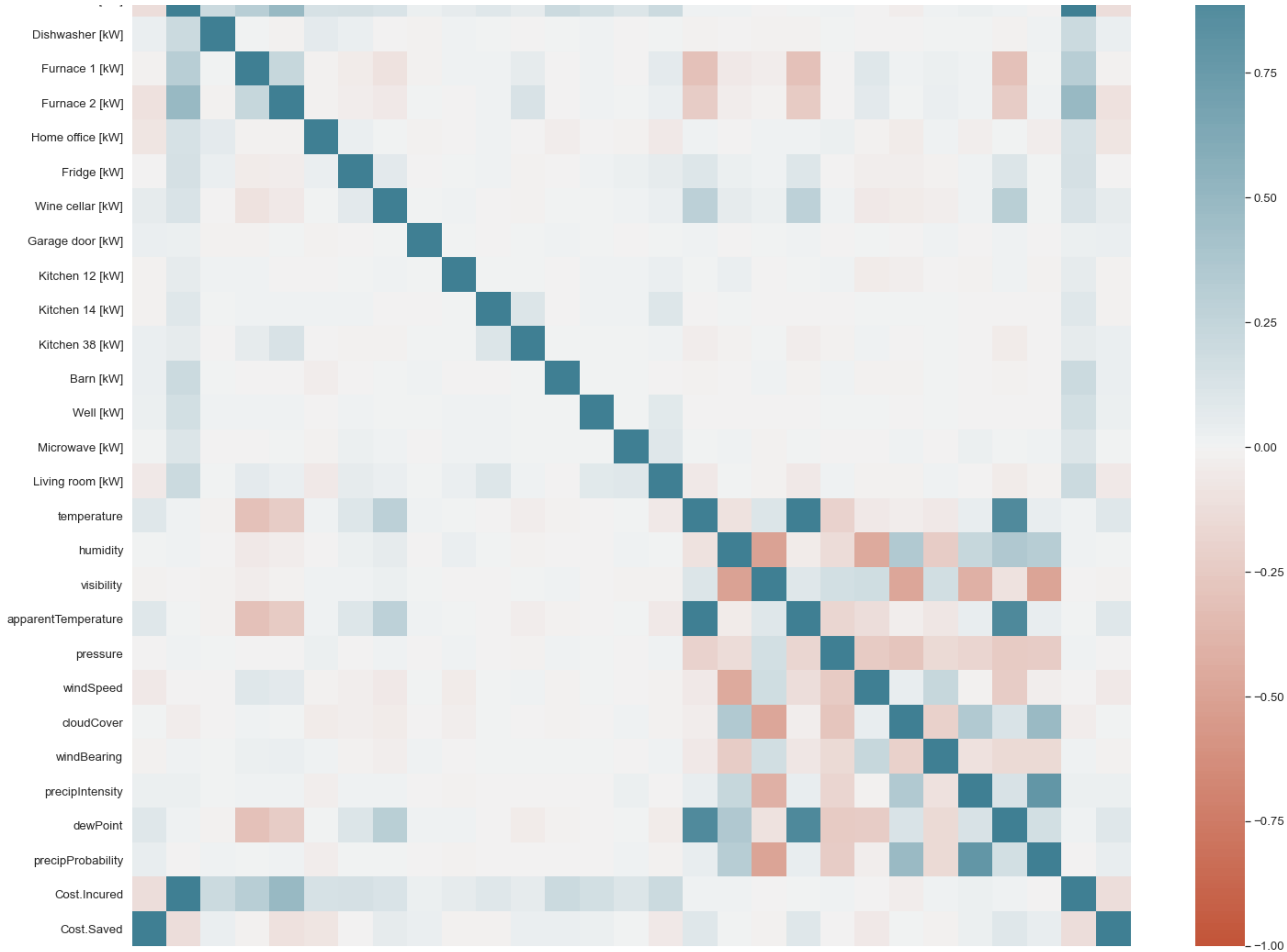
Correlation

In [17]:

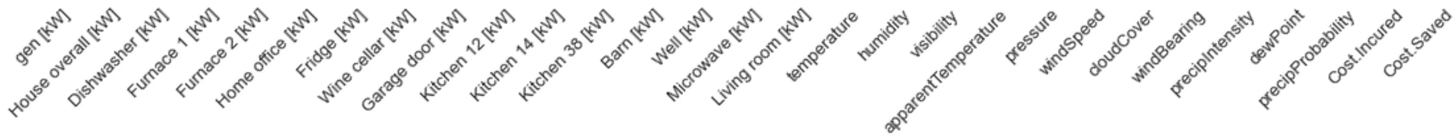
```
import seaborn as sns
sns.set(color_codes=True, font_scale=1.2)
test_df = df.copy()
test_df.drop('time', axis=1, inplace=True)
test_df.drop('summary', axis=1, inplace=True)
corr = test_df.corr()

#corr.style.background_gradient(cmap='coolwarm')
pyplot.figure(figsize = (25,20))
ax = sns.heatmap(
    corr,
    vmin=-1, vmax=1, center=0,
    cmap=sns.diverging_palette(20, 220, n=200),
    square=True
)
ax.set_xticklabels(
    ax.get_xticklabels(),
    rotation=45,
    horizontalalignment='right'
);
```





Dishwasher [kW]
Furnace 1 [kW]
Furnace 2 [kW]
Home office [kW]
Fridge [kW]
Wine cellar [kW]
Garage door [kW]
Kitchen 12 [kW]
Kitchen 14 [kW]
Kitchen 38 [kW]
Barn [kW]
Well [kW]
Microwave [kW]
Living room [kW]
temperature
humidity
visibility
apparentTemperature
pressure
windSpeed
cloudCover
windBearing
precipIntensity
dewPoint
precipProbability
Cost.Incured
Cost.Saved
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00



T-Test

In [18]:

```
types_map = df.dtypes.to_dict()
num_columns = [
'Dishwasher [kW]',
'Furnace 1 [kW]',
'Furnace 2 [kW]',
'Home office [kW]',
'Fridge [kW]',
'Wine cellar [kW]',
'Garage door [kW]',
'Kitchen 12 [kW]',
'Kitchen 14 [kW]',
'Kitchen 38 [kW]',
'Barn [kW]',
'Well [kW]',
'Microwave [kW]',
'Living room [kW]']


print(num_columns)

for i in range(len(num_columns)-1):
    for j in range(i+1,len(num_columns)):
        col1 = num_columns[i]
        col2 = num_columns[j]
        t_val, p_val = stats.ttest_ind(df[col1], df[col2])
        print("(%s,%s) => t-value=%s, p-value=%s" % (num_columns[i], num_columns[j], str(t_val), str(p_val)))
```

```
['Dishwasher [kW]', 'Furnace 1 [kW]', 'Furnace 2 [kW]', 'Home office [kW]', 'Fridge [kW]', 'Wine cellar [kW]', 'Garag
e door [kW]', 'Kitchen 12 [kW]', 'Kitchen 14 [kW]', 'Kitchen 38 [kW]', 'Barn [kW]', 'Well [kW]', 'Microwave [kW]', 'L
iving room [kW]']
(Dishwasher [kW],Furnace 1 [kW]) => t-value=-188.83322552742646, p-value=0.0
(Dishwasher [kW],Furnace 2 [kW]) => t-value=-286.1721793337311, p-value=0.0
(Dishwasher [kW],Home office [kW]) => t-value=-162.8052306239252, p-value=0.0
```

```
(Dishwasher [kW],Fridge [kW]) => t-value=-111.14078014432731, p-value=0.0
(Dishwasher [kW],Wine cellar [kW]) => t-value=-38.30824517058043, p-value=0.0
(Dishwasher [kW],Garage door [kW]) => t-value=63.86837358121533, p-value=0.0
(Dishwasher [kW],Kitchen 12 [kW]) => t-value=105.67630850369983, p-value=0.0
(Dishwasher [kW],Kitchen 14 [kW]) => t-value=83.97351980175974, p-value=0.0
(Dishwasher [kW],Kitchen 38 [kW]) => t-value=116.57472346760788, p-value=0.0
(Dishwasher [kW],Barn [kW]) => t-value=-69.23756195423603, p-value=0.0
(Dishwasher [kW],Well [kW]) => t-value=47.40032941642572, p-value=0.0
(Dishwasher [kW],Microwave [kW]) => t-value=67.29607516648211, p-value=0.0
(Dishwasher [kW],Living room [kW]) => t-value=-13.102306946990907, p-value=3.217701520268452e-39
(Furnace 1 [kW],Furnace 2 [kW]) => t-value=-108.43317300528061, p-value=0.0
(Furnace 1 [kW],Home office [kW]) => t-value=64.02214872063885, p-value=0.0
(Furnace 1 [kW],Fridge [kW]) => t-value=136.4848788562009, p-value=0.0
(Furnace 1 [kW],Wine cellar [kW]) => t-value=226.69204677107444, p-value=0.0
(Furnace 1 [kW],Garage door [kW]) => t-value=355.93732537222354, p-value=0.0
(Furnace 1 [kW],Kitchen 12 [kW]) => t-value=401.66275614760286, p-value=0.0
(Furnace 1 [kW],Kitchen 14 [kW]) => t-value=352.47347510279246, p-value=0.0
(Furnace 1 [kW],Kitchen 38 [kW]) => t-value=416.5366069116952, p-value=0.0
(Furnace 1 [kW],Barn [kW]) => t-value=109.40561194408048, p-value=0.0
(Furnace 1 [kW],Well [kW]) => t-value=271.95689169287465, p-value=0.0
(Furnace 1 [kW],Microwave [kW]) => t-value=319.79642423279967, p-value=0.0
(Furnace 1 [kW],Living room [kW]) => t-value=233.27554110411083, p-value=0.0
(Furnace 2 [kW],Home office [kW]) => t-value=190.35917571294385, p-value=0.0
(Furnace 2 [kW],Fridge [kW]) => t-value=267.6475471529655, p-value=0.0
(Furnace 2 [kW],Wine cellar [kW]) => t-value=357.737128112753, p-value=0.0
(Furnace 2 [kW],Garage door [kW]) => t-value=485.80891924301, p-value=0.0
(Furnace 2 [kW],Kitchen 12 [kW]) => t-value=528.6559064703271, p-value=0.0
(Furnace 2 [kW],Kitchen 14 [kW]) => t-value=473.7716102789579, p-value=0.0
(Furnace 2 [kW],Kitchen 38 [kW]) => t-value=543.5117741371503, p-value=0.0
(Furnace 2 [kW],Barn [kW]) => t-value=205.58914669855062, p-value=0.0
(Furnace 2 [kW],Well [kW]) => t-value=381.1140832133386, p-value=0.0
(Furnace 2 [kW],Microwave [kW]) => t-value=437.38963882068623, p-value=0.0
(Furnace 2 [kW],Living room [kW]) => t-value=355.13019854947987, p-value=0.0
(Home office [kW],Fridge [kW]) => t-value=97.33889818914413, p-value=0.0
(Home office [kW],Wine cellar [kW]) => t-value=232.62023591806715, p-value=0.0
(Home office [kW],Garage door [kW]) => t-value=452.06982247599257, p-value=0.0
(Home office [kW],Kitchen 12 [kW]) => t-value=522.3229946799493, p-value=0.0
(Home office [kW],Kitchen 14 [kW]) => t-value=406.6971899508597, p-value=0.0
(Home office [kW],Kitchen 38 [kW]) => t-value=552.29494810826, p-value=0.0
(Home office [kW],Barn [kW]) => t-value=70.84078798151572, p-value=0.0
(Home office [kW],Well [kW]) => t-value=269.42908070719386, p-value=0.0
(Home office [kW],Microwave [kW]) => t-value=346.98845665072855, p-value=0.0
(Home office [kW],Living room [kW]) => t-value=229.9639039939466, p-value=0.0
(Fridge [kW],Wine cellar [kW]) => t-value=158.8148384311214, p-value=0.0
(Fridge [kW],Garage door [kW]) => t-value=452.4810532742612, p-value=0.0
```

```
(Fridge [kW],Kitchen 12 [kW]) => t-value=544.4613678553065, p-value=0.0
(Fridge [kW],Kitchen 14 [kW]) => t-value=371.0882371533266, p-value=0.0
(Fridge [kW],Kitchen 38 [kW]) => t-value=592.0039009577309, p-value=0.0
(Fridge [kW],Barn [kW]) => t-value=16.477967009501, p-value=5.381389397118558e-61
(Fridge [kW],Well [kW]) => t-value=215.9535215959703, p-value=0.0
(Fridge [kW],Microwave [kW]) => t-value=298.99837628689653, p-value=0.0
(Fridge [kW],Living room [kW]) => t-value=163.52075774127889, p-value=0.0
(Wine cellar [kW],Garage door [kW]) => t-value=332.8891981456782, p-value=0.0
(Wine cellar [kW],Kitchen 12 [kW]) => t-value=451.24473925121066, p-value=0.0
(Wine cellar [kW],Kitchen 14 [kW]) => t-value=259.1792142998726, p-value=0.0
(Wine cellar [kW],Kitchen 38 [kW]) => t-value=515.8906593716237, p-value=0.0
(Wine cellar [kW],Barn [kW]) => t-value=-55.19460543701042, p-value=0.0
(Wine cellar [kW],Well [kW]) => t-value=125.77510642736752, p-value=0.0
(Wine cellar [kW],Microwave [kW]) => t-value=192.97424339763592, p-value=0.0
(Wine cellar [kW],Living room [kW]) => t-value=43.17615964817937, p-value=0.0
(Garage door [kW],Kitchen 12 [kW]) => t-value=309.41357706010206, p-value=0.0
(Garage door [kW],Kitchen 14 [kW]) => t-value=64.71282920218825, p-value=0.0
(Garage door [kW],Kitchen 38 [kW]) => t-value=701.8119640875567, p-value=0.0
(Garage door [kW],Barn [kW]) => t-value=-155.06790224975617, p-value=0.0
(Garage door [kW],Well [kW]) => t-value=-7.698387260774723, p-value=1.3791835926998015e-14
(Garage door [kW],Microwave [kW]) => t-value=22.429773901868057, p-value=2.14775212778068e-111
(Garage door [kW],Living room [kW]) => t-value=-154.7718545273762, p-value=0.0
(Kitchen 12 [kW],Kitchen 14 [kW]) => t-value=-37.967791374369305, p-value=0.0
(Kitchen 12 [kW],Kitchen 38 [kW]) => t-value=89.17261461276885, p-value=0.0
(Kitchen 12 [kW],Barn [kW]) => t-value=-194.1923173289999, p-value=0.0
(Kitchen 12 [kW],Well [kW]) => t-value=-65.54566506008857, p-value=0.0
(Kitchen 12 [kW],Microwave [kW]) => t-value=-57.68606916717358, p-value=0.0
(Kitchen 12 [kW],Living room [kW]) => t-value=-234.6046594693967, p-value=0.0
(Kitchen 14 [kW],Kitchen 38 [kW]) => t-value=64.87928597996128, p-value=0.0
(Kitchen 14 [kW],Barn [kW]) => t-value=-168.68917687509952, p-value=0.0
(Kitchen 14 [kW],Well [kW]) => t-value=-38.78137533734248, p-value=0.0
(Kitchen 14 [kW],Microwave [kW]) => t-value=-22.461335445893873, p-value=1.056497783430865e-111
(Kitchen 14 [kW],Living room [kW]) => t-value=-163.3388430848913, p-value=0.0
(Kitchen 38 [kW],Barn [kW]) => t-value=-204.93444220419005, p-value=0.0
(Kitchen 38 [kW],Well [kW]) => t-value=-80.50619089642245, p-value=0.0
(Kitchen 38 [kW],Microwave [kW]) => t-value=-78.79723166960207, p-value=0.0
(Kitchen 38 [kW],Living room [kW]) => t-value=-260.8958547109183, p-value=0.0
(Barn [kW],Well [kW]) => t-value=124.19568273500262, p-value=0.0
(Barn [kW],Microwave [kW]) => t-value=149.65634155508673, p-value=0.0
(Barn [kW],Living room [kW]) => t-value=73.47204387848988, p-value=0.0
(Well [kW],Microwave [kW]) => t-value=19.497361230261482, p-value=1.1983473218012784e-84
(Well [kW],Living room [kW]) => t-value=-83.11240045155618, p-value=0.0
(Microwave [kW],Living room [kW]) => t-value=-125.29673511674937, p-value=0.0
```

It can be seen from the results that all the ttest are significant.

Exporting Transformed data into csv.

In [19]:
```
df.to_csv('smart.clean.csv')
```