Report Code (Zhongyuan Li, Shuo Li)

December 9, 2016

In [144]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn import tree
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import KFold
%matplotlib inline
```

In [145]:
```
text = np.genfromtxt('MIC_results.txt',delimiter='/n',dtype='float')
selection = []
selectIndex = 0.2
value = []

for i in range(0,839):
    if text[i] > selectIndex:
        selection.append(i)
        value.append(text[i])
print(selection)
print(value)
```

```
[3, 4, 21, 22, 25, 26, 29, 33, 34, 37, 39, 42, 45, 313, 460, 589, 600, 608, 826, 82
[0.209046706838, 0.20617031973399999, 0.205602222833, 0.205602222833, 0.21419561692
```

In [146]:
```
csv = open('recs2009_public.csv','rb')
data = pd.read_csv(csv)
csv.close()
data = data.drop(['DOEID'],axis=1)
data.head()
```

Out[146]:

	REGIONC	DIVISION	REPORTABLE_DOMAIN	TYPEHUQ	NWEIGHT	HDD65	CD
0	2	4	12	2	2471.679705	4742	1
1	4	10	26	2	8599.172010	2662	
2	1	1	1	5	8969.915921	6233	
3	2	3	7	2	18003.639600	6034	
4	1	1	1	3	5999.605242	5388	

	HDD30YR	CDD30YR	Climate_Region_Pub	...	SCALEEL	KAVALNG	PERIOD

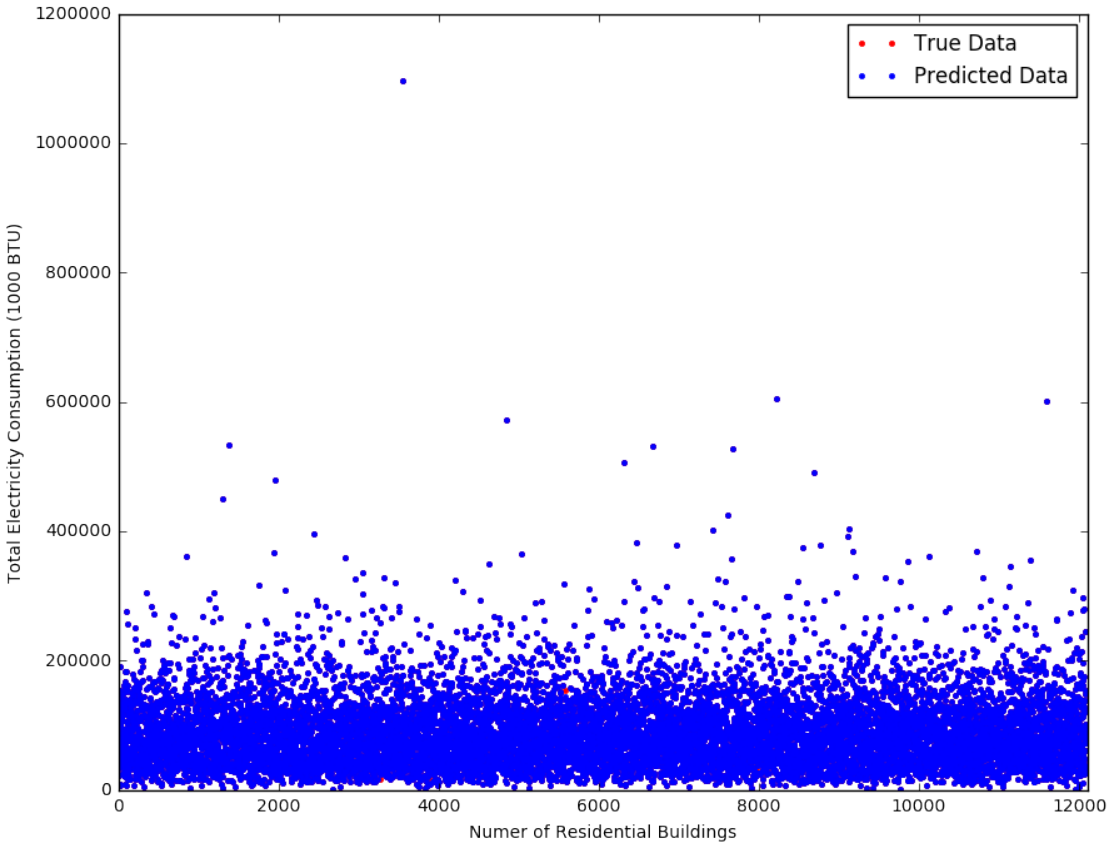
```
0     4953     1271                    4    ...            0       -2
1     2688      143                    5    ...            0        1
2     5741      829                    1    ...            0        3
3     5781      868                    1    ...            3        3
4     5313      797                    1    ...            0        1

   SCALENG  PERIODLP  SCALELP  PERIODFO  SCALEFO  PERIODKR  SCALEKER
0       -2        -2       -2        -2       -2        -2        -2
1        0        -2       -2        -2       -2        -2        -2
2        3        -2       -2        -2       -2        -2        -2
3        3        -2       -2        -2       -2        -2        -2
4        0        -2       -2        -2       -2        -2        -2

[5 rows x 930 columns]
```

In [147]:
```
data[data.columns[906]]
data.columns[906]
```

Out[147]:
```
'TOTALBTU'
```

In [148]:
```
#### SelectIndex = 0.20, select by computer ########

new = pd.DataFrame()
Y = data[data.columns[906]]
Y = pd.DataFrame(Y)
for i in selection:
    new[data.columns[i]] = data[data.columns[i]]
X = new
reg1 = tree.DecisionTreeRegressor()
reg1.fit(X,Y)
print(reg1.score(X,Y))

fig1 = plt.figure(figsize=(10,8))
plt.plot(Y,'.',c='r',label='True Data')
plt.xlim(0,12100)
plt.ylabel('Total Electricity Consumption (1000 BTU)')
plt.xlabel('Numer of Residential Buildings')
plt.plot(reg1.predict(X),'.',c='b',label='Predicted Data')
plt.legend()
plt.savefig('books_read.png')
```

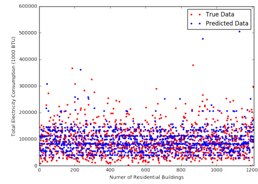
```
0.999932251406
```

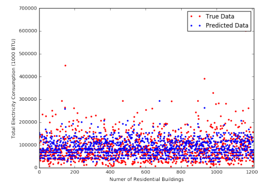


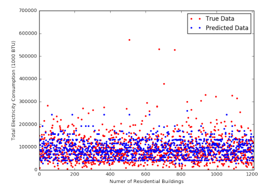

```
In [149]: X1 = X.as_matrix()
          Y1 = Y.as_matrix()

          kf = KFold(n_splits=10,shuffle=True,random_state=True)
          sum1 = []
          i = 0
          fig = plt.figure(figsize=(8,70))
          for train, test in kf.split(X1):
              X_train = X1[train]
              X_test = X1[test]
              Y_train = Y1[train]
              Y_test = Y1[test]
              regr_1 = DecisionTreeRegressor(max_depth=7)
              regr_1.fit(X_train, Y_train)
              sum1.append(regr_1.score(X_test,Y_test))
              plt.subplot(10,1,i+1)
              plt.plot(Y_test,'.',c='r',label='True Data')
              plt.plot(regr_1.predict(X_test),'.',c='b',label='Predicted Data')
              plt.xlim(0,1210)
              plt.ylabel('Total Electricity Consumption (1000 BTU)')
```

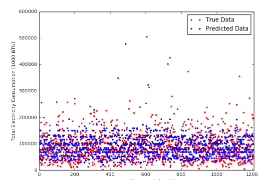
```
            plt.xlabel('Numer of Residential Buildings')
            plt.legend()
            plt.savefig(str(i)+'.png')
            i = i+1
        sum1 = np.array(sum1)
        print(sum1)
        print(sum1.mean())
```

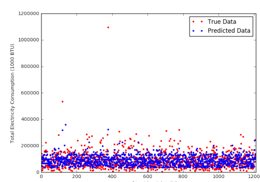
```
[ 0.29036773  0.39202908  0.37182026  0.33236584  0.42443807  0.31279686
  0.39862753  0.38248166  0.32020073  0.31118168]
0.353630944166
```

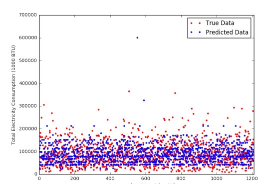

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

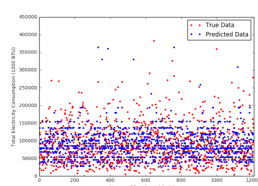

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

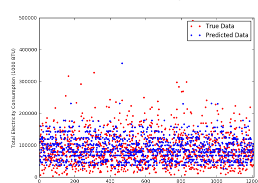

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

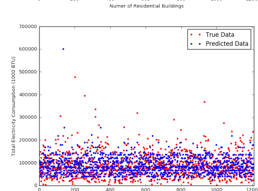

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

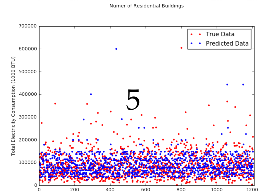

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings


True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings


True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings


True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

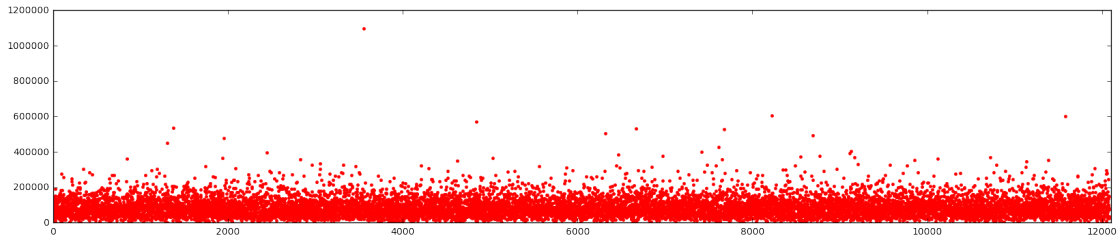

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings


True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

In [150]:
```
###### SelectIndex = 0.24 ########
selection1 = []
selectIndex = 0.24
for i in range(0,839):
    if text[i] > selectIndex:
        selection1.append(i)

new = pd.DataFrame()
Y = data[data.columns[906]]
Y = pd.DataFrame(Y)
for i in selection1:
    new[data.columns[i]] = data[data.columns[i]]
X = new
reg1 = tree.DecisionTreeRegressor()
reg1.fit(X,Y)
print(reg1.score(X,Y))

fig1 = plt.figure(figsize=(20,4))
plt.plot(Y,'.',c='r')
plt.xlim(0,12100)
```

```
0.954232225921
```

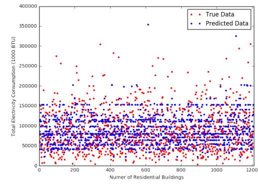
Out[150]: (0, 12100)

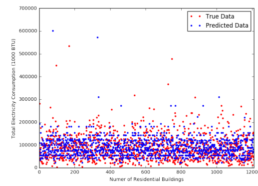


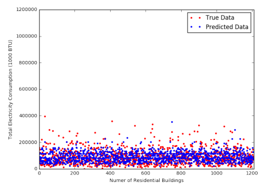
In [151]:
```
X1 = X.as_matrix()
Y1 = Y.as_matrix()

kf = KFold(n_splits=10,shuffle=False,random_state=None)
sum1 = []
i = 0
fig = plt.figure(figsize=(8,70))
for train, test in kf.split(X1):
    X_train = X1[train]
```

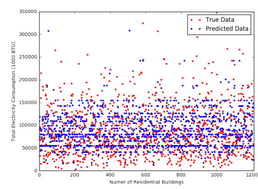
```
            X_test = X1[test]
            Y_train = Y1[train]
            Y_test = Y1[test]
            regr_1 = DecisionTreeRegressor(max_depth=7)
            regr_1.fit(X_train, Y_train)
            sum1.append(regr_1.score(X_test,Y_test))
            plt.subplot(10,1,i+1)
            plt.plot(Y_test,'.',c='r',label='True Data')
            plt.plot(regr_1.predict(X_test),'.',c='b',label='Predicted Data')
            plt.xlim(0,1210)
            plt.ylabel('Total Electricity Consumption (1000 BTU)')
            plt.xlabel('Numer of Residential Buildings')
            plt.legend()
            plt.savefig('99.png')
            i = i+1
        sum1 = np.array(sum1)
        print(sum1)
        print(sum1.mean())
```

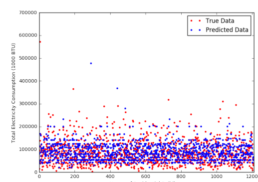
```
[ 0.34231338  0.27263038  0.38287027  0.40756511  0.25278089  0.42463847
  0.37635837  0.39572472  0.34888857  0.2696346 ]
0.347340477441
```

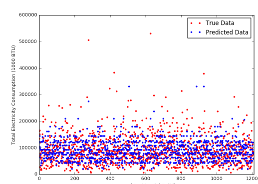

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

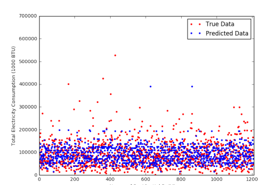

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

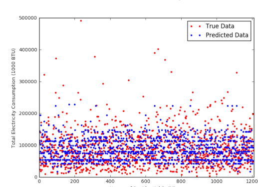

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

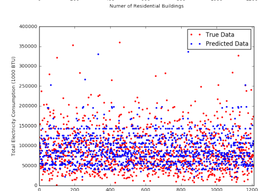

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

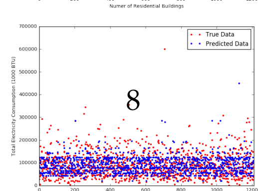

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings


True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings


True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings


True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

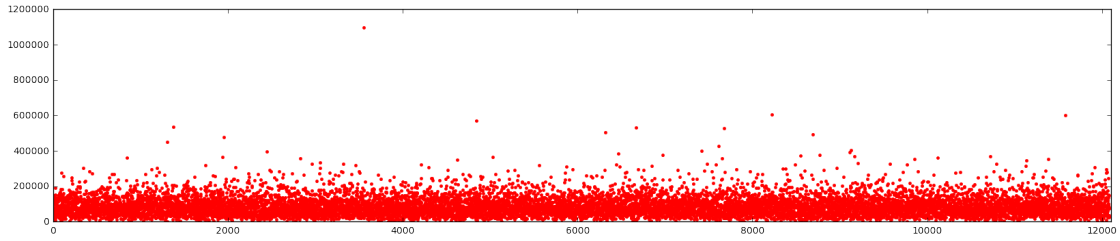

True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings


True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

In [152]:
```
###### Select by hand ########
selection2=[3,26,33,34,460,600,608,826,827,828]
new = pd.DataFrame()
Y = data[data.columns[906]]
Y = pd.DataFrame(Y)
for i in selection2:
    new[data.columns[i]] = data[data.columns[i]]
X = new
reg1 = tree.DecisionTreeRegressor()
reg1.fit(X,Y)
print(reg1.score(X,Y))

fig1 = plt.figure(figsize=(20,4))
plt.plot(Y,'.',c='r')
plt.xlim(0,12100)
```

```
0.987371774421
```

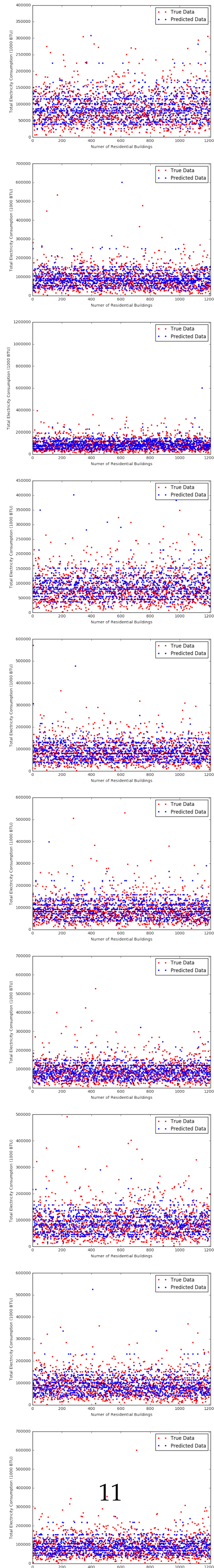
Out[152]: (0, 12100)



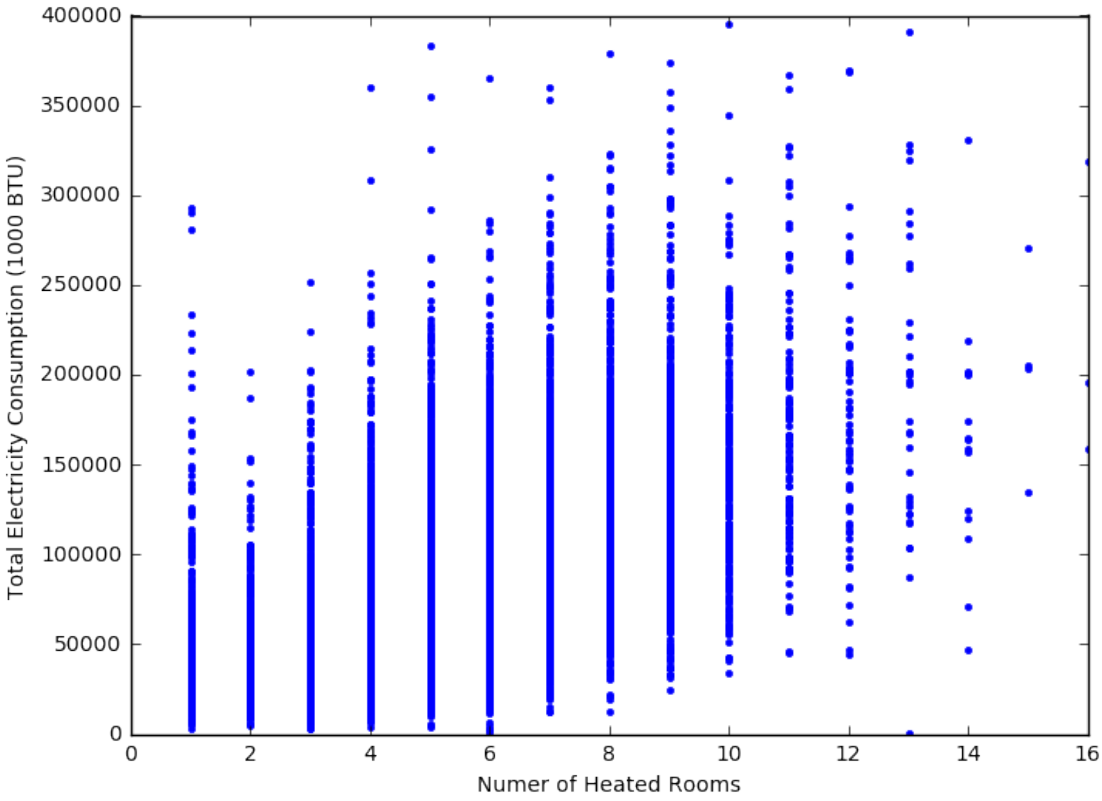
In [153]:
```
X1 = X.as_matrix()
Y1 = Y.as_matrix()

kf = KFold(n_splits=10,shuffle=False,random_state=None)
sum1 = []
i = 0
fig = plt.figure(figsize=(8,70))
for train, test in kf.split(X1):
    X_train = X1[train]
    X_test = X1[test]
    Y_train = Y1[train]
    Y_test = Y1[test]
    regr_1 = DecisionTreeRegressor(max_depth=7)
    regr_1.fit(X_train, Y_train)
```

```
            sum1.append(regr_1.score(X_test,Y_test))
            plt.subplot(10,1,i+1)
            plt.plot(Y_test,'.',c='r',label='True Data')
            plt.plot(regr_1.predict(X_test),'.',c='b',label='Predicted Data')
            plt.xlim(0,1210)
            plt.ylabel('Total Electricity Consumption (1000 BTU)')
            plt.xlabel('Numer of Residential Buildings')
            plt.legend()
            plt.savefig('999.png')
            i = i+1
        sum1 = np.array(sum1)
        print(sum1)
        print(sum1.mean())
```

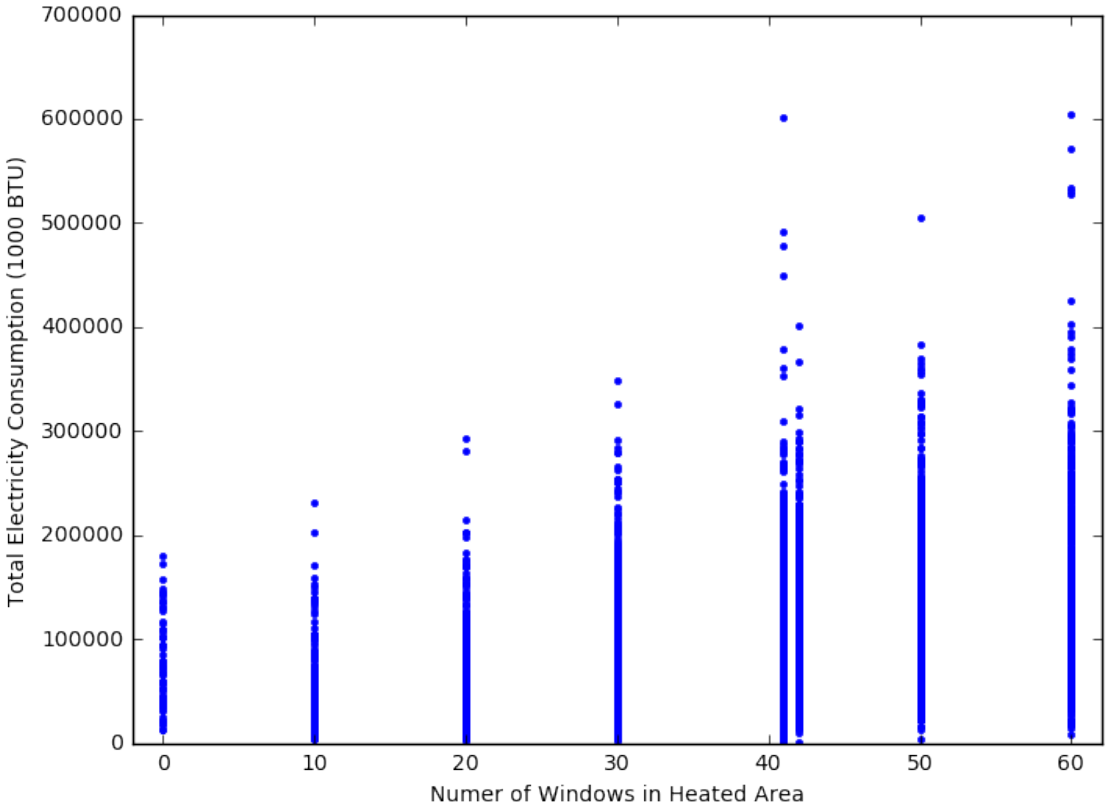
```
[ 0.36739031  0.31843039  0.32467625  0.3825026   0.41423312  0.37781657
  0.36939824  0.39436142  0.28747249  0.30829111]
0.354457249117
```


True Data
Predicted Data
Total Electricity Consumption (1000 BTU)
Numer of Residential Buildings

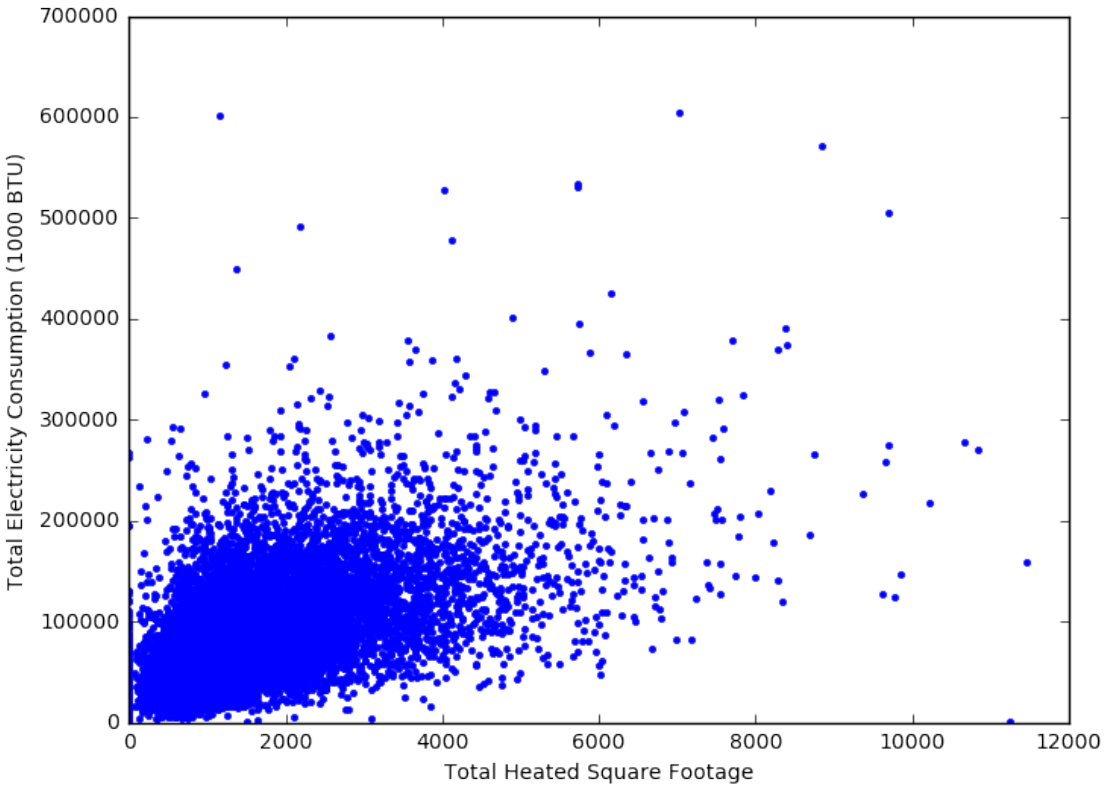
In [154]:
```
###### Single Feature Analysis: Heatroom #####
new
fig = plt.figure(figsize=(8,6))
plt.plot(new.HEATROOM,Y.TOTALBTU,'.')
plt.ylabel('Total Electricity Consumption (1000 BTU)')
plt.xlabel('Numer of Heated Rooms')
plt.ylim(0,400000)
plt.xlim(0,16)
plt.savefig('Heatroom.png')
```



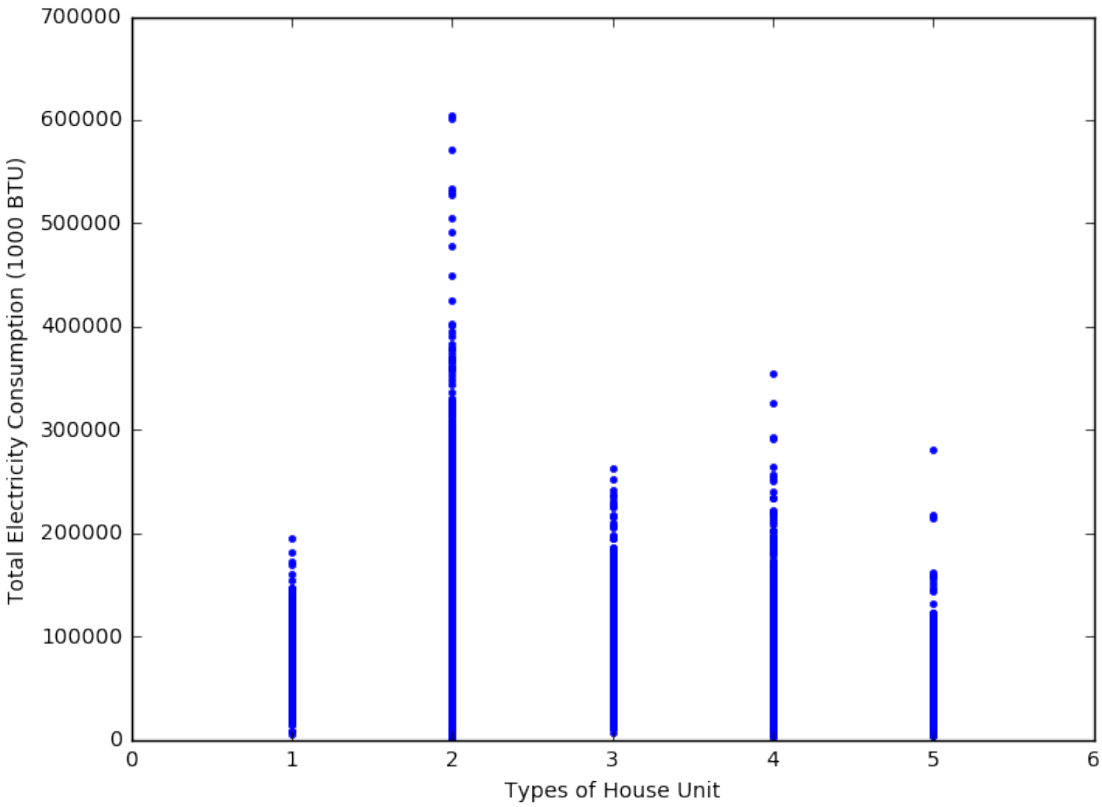
In [155]:
```
##### Single Feature Analysis: Numer of Windows in Heated Area #####
fig = plt.figure(figsize=(8,6))
plt.plot(new.WINDOWS,Y.TOTALBTU,'.')
plt.ylabel('Total Electricity Consumption (1000 BTU)')
plt.xlabel('Numer of Windows in Heated Area')
plt.ylim(0,700000)
plt.xlim(-2,62)
plt.savefig('Windows.png')
```



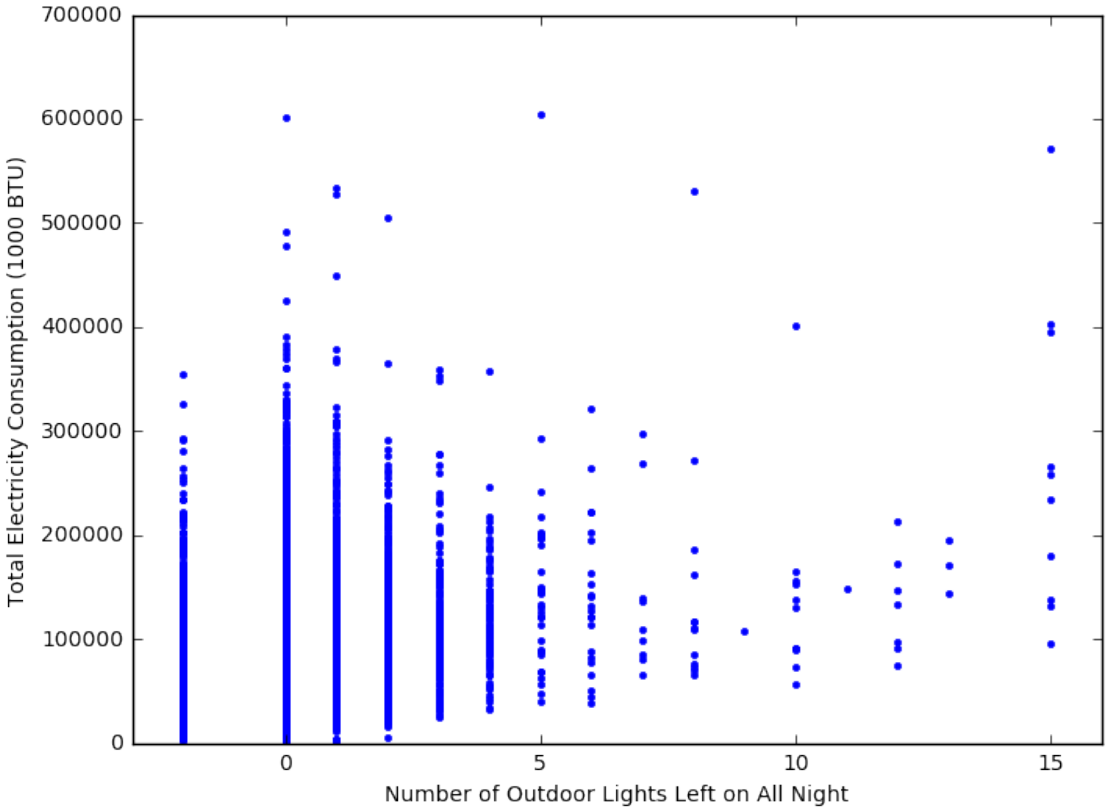

```
In [156]: ##### Single Feature Analysis: Total Heated Squre Footage #####
          fig = plt.figure(figsize=(8,6))
          plt.plot(new.TOTHSQFT,Y.TOTALBTU,'.')
          plt.ylabel('Total Electricity Consumption (1000 BTU)')
          plt.xlabel('Total Heated Square Footage')
          plt.ylim(0,700000)
          plt.xlim(-10,12000)
          plt.savefig('Square Footage.png')
```



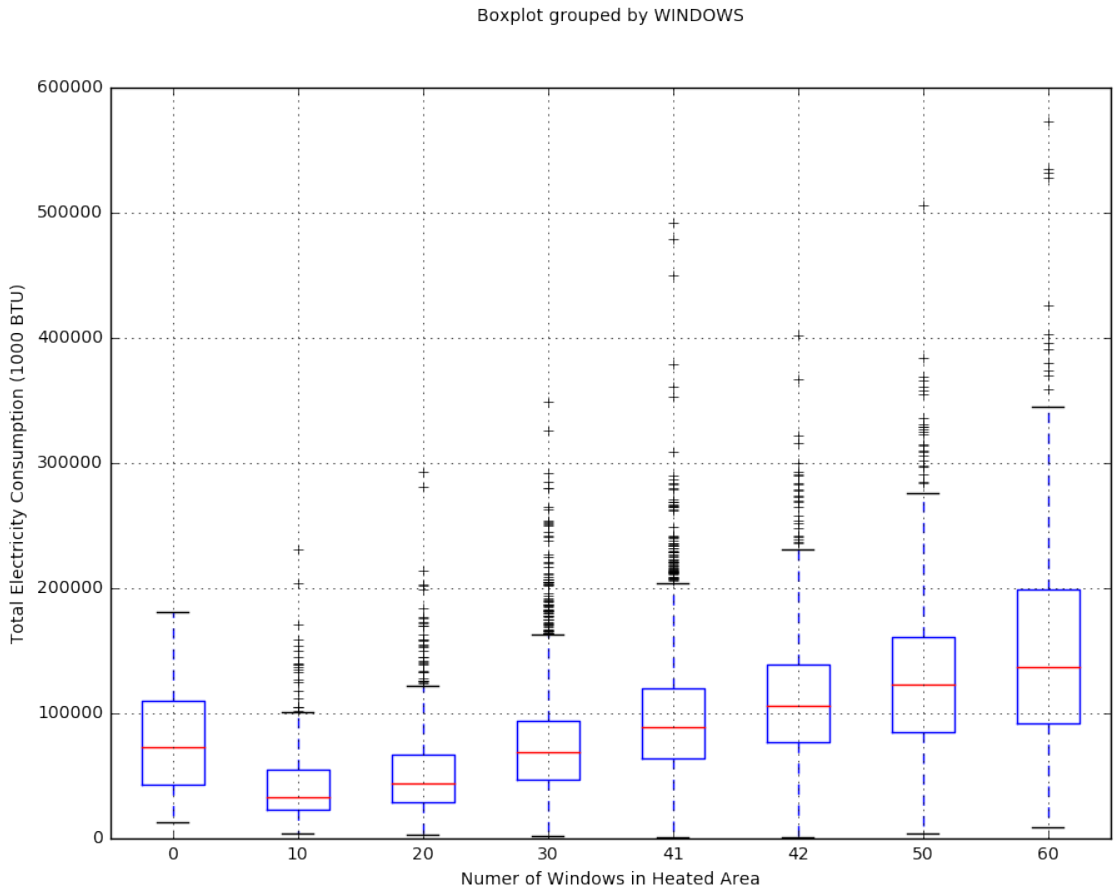

```
In [157]: ##### Single Feature Analysis: Types of House Unit #####
          fig = plt.figure(figsize=(8,6))
          plt.plot(new.TYPEHUQ,Y.TOTALBTU,'.')
          plt.ylabel('Total Electricity Consumption (1000 BTU)')
          plt.xlabel('Types of House Unit')
          plt.ylim(0,700000)
          plt.xlim(0,6)
          plt.savefig('Types of House Unit.png')
```



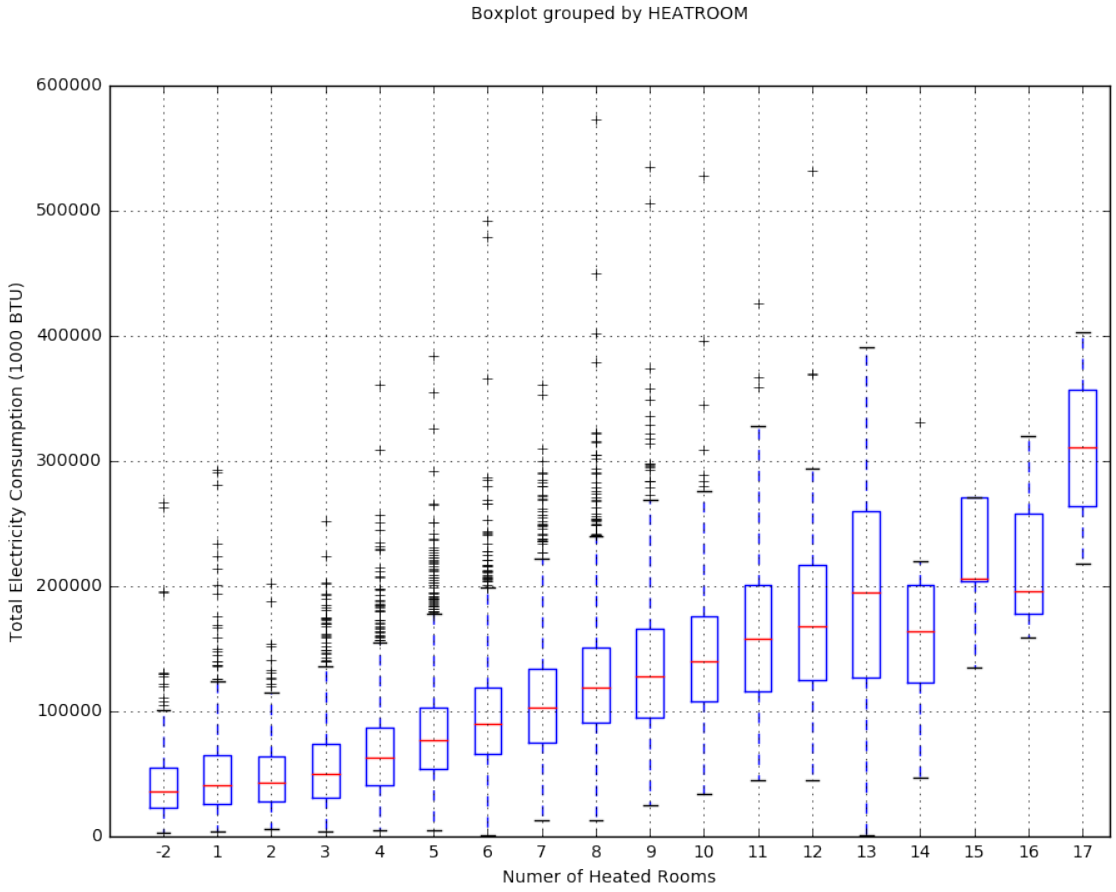
In [158]:
```
##### Single Feature Analysis: Number of Outdoor Lights Left on All Night
fig = plt.figure(figsize=(8,6))
plt.plot(new.NOUTLGTNT,Y.TOTALBTU,'.')
plt.ylabel('Total Electricity Consumption (1000 BTU)')
plt.xlabel('Number of Outdoor Lights Left on All Night')
plt.ylim(0,700000)
plt.xlim(-3,16)
plt.savefig('Outdoor Lights.png')
```



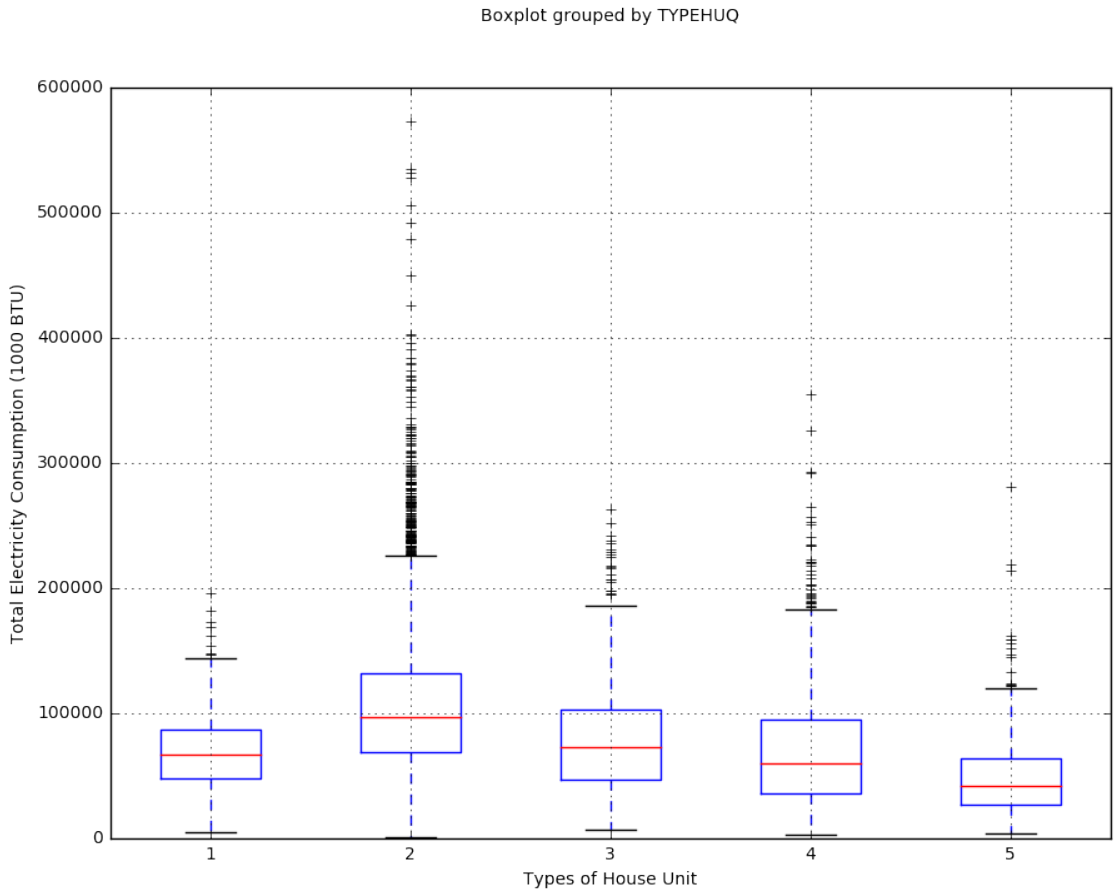

```
In [159]: ###### Single Feature Analysis: Windows #####
          new1 = new
          new1['VALUE'] = Y.TOTALBTU
          new1.boxplot(column='VALUE',by='WINDOWS',return_type='axes',figsize=(10,8
          plt.ylabel('Total Electricity Consumption (1000 BTU)')
          plt.xlabel('Numer of Windows in Heated Area')
          plt.ylim(0,600000)
          plt.title('')
          plt.savefig('Windows1.png')
```



In [160]:
```
###### Single Feature Analysis: HEAT ROOM #####
new1.boxplot(column='VALUE',by='HEATROOM',return_type='axes',figsize=(10,
plt.ylabel('Total Electricity Consumption (1000 BTU)')
plt.xlabel('Numer of Heated Rooms')
plt.title('')
plt.xlim(0,18.5)
plt.ylim(0,600000)
plt.savefig('Heatroom1.png')
```




```
In [161]: ##### Single Feature Analysis: Types of House Unit #####
          new1.boxplot(column='VALUE',by='TYPEHUQ',return_type='axes',figsize=(10,8
          plt.ylabel('Total Electricity Consumption (1000 BTU)')
          plt.xlabel('Types of House Unit')
          plt.title('')
          plt.ylim(0,600000)
          plt.savefig('Types of House Unit1.png')
```



In [162]:
```
##### Single Feature Analysis: Number of Outdoor Lights Left on All Night
new1.boxplot(column='VALUE',by='NOUTLGTNT',return_type='axes',figsize=(10
plt.ylabel('Total Electricity Consumption (1000 BTU)')
plt.xlabel('Number of Outdoor Lights Left on All Night')
plt.ylim(0,600000)
plt.title('')
plt.savefig('Outdoor Lights1.png')
new1
```

Out[162]:

	TYPEHUQ	NAPTFLRS	TOTROOMS	CELLAR	HEATROOM	NOUTLGTNT	WINDOWS
0	2	-2	9	1	9	0	41
1	2	-2	4	0	4	0	41
2	5	1	2	-2	2	-2	20
3	2	-2	7	0	7	0	41
4	3	-2	5	1	5	0	30
5	2	-2	6	1	6	1	30
6	2	-2	7	1	7	2	50
7	2	-2	6	0	6	0	20
8	3	-2	7	1	7	0	42

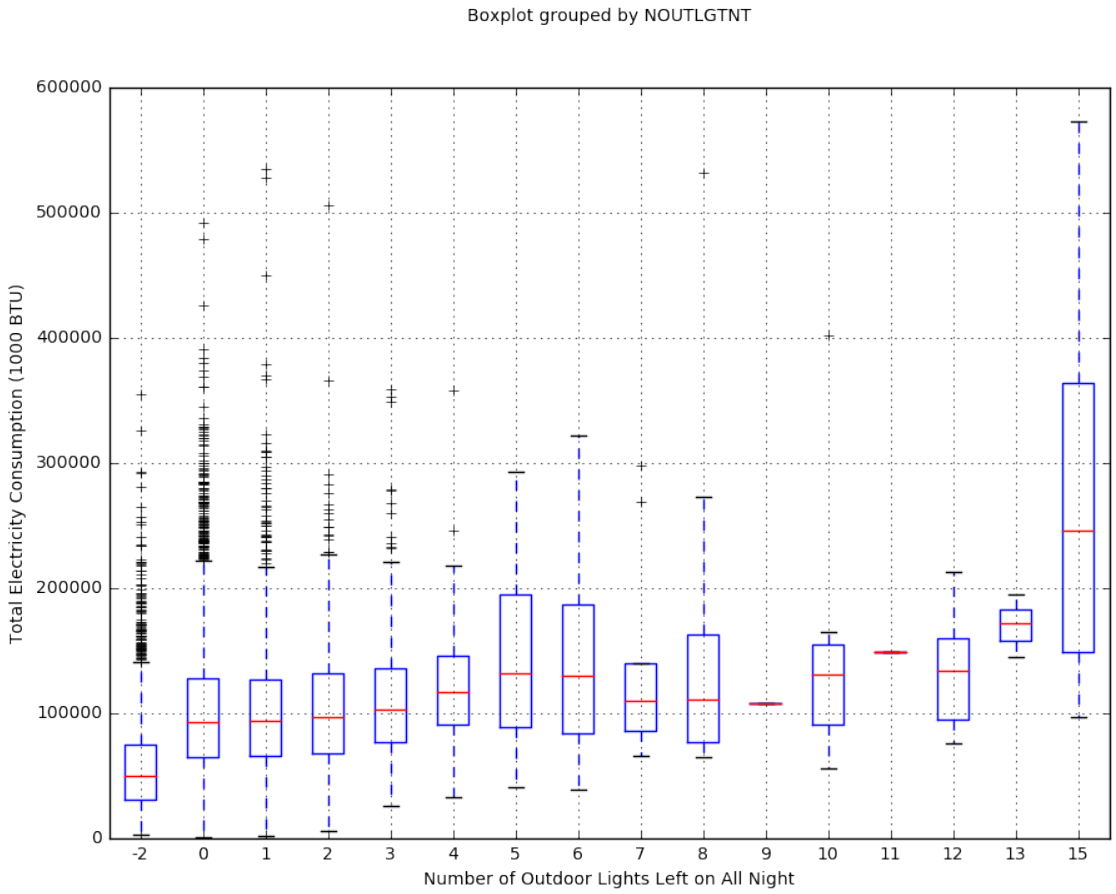
9	2	-2	7	1	7	0	41
10	4	1	3	0	1	-2	20
11	2	-2	7	1	7	0	50
12	2	-2	6	0	-2	0	42
13	5	1	4	-2	4	-2	10
14	5	1	1	-2	1	-2	10
15	2	-2	9	1	9	0	42
16	2	-2	6	0	-2	0	41
17	4	1	3	1	3	-2	41
18	1	-2	5	-2	5	-2	41
19	2	-2	12	1	12	0	42
20	2	-2	6	0	6	1	30
21	2	-2	6	0	6	0	30
22	2	-2	4	0	4	1	42
23	2	-2	8	0	8	0	41
24	2	-2	9	1	9	0	41
25	4	1	5	1	5	-2	41
26	5	2	5	-2	5	-2	41
27	2	-2	4	1	1	0	30
28	1	-2	2	-2	-2	-2	60
29	2	-2	7	0	4	2	41
...	...	...	...	...	...	...	...
12053	5	1	4	-2	4	-2	20
12054	2	-2	7	0	7	0	42
12055	2	-2	6	0	6	0	41
12056	2	-2	5	1	5	1	30
12057	5	1	5	-2	4	-2	41
12058	2	-2	5	0	5	0	41
12059	2	-2	7	1	7	1	30
12060	2	-2	7	0	7	0	50
12061	2	-2	6	1	6	0	50
12062	2	-2	6	0	6	0	50
12063	2	-2	8	1	8	1	41
12064	2	-2	3	0	3	0	30
12065	2	-2	5	1	5	0	41
12066	5	1	4	-2	4	-2	30
12067	5	1	1	-2	1	-2	20
12068	5	1	3	-2	2	-2	30
12069	2	-2	6	1	6	0	41
12070	3	-2	5	0	5	0	30
12071	2	-2	10	1	10	4	50
12072	2	-2	7	0	2	1	42
12073	2	-2	9	0	4	0	50
12074	2	-2	6	0	6	0	30
12075	2	-2	4	0	4	2	41
12076	3	-2	3	0	3	0	20
12077	2	-2	7	0	7	0	50
12078	2	-2	6	1	5	0	30

12079	4	1	1	0	1	-2	30
12080	2	-2	8	0	8	0	42
12081	2	-2	5	1	5	0	41
12082	2	-2	4	1	4	0	30

	TOTSQFT	TOTSQFT_EN	TOTHSQFT	VALUE
0	5075	4675	3958	63006
1	3136	2736	2736	103460
2	528	528	528	58716
3	2023	1623	1623	76401
4	1912	1912	1274	59809
5	3485	3485	3485	114350
6	2654	2654	2296	150726
7	2352	1952	1952	78230
8	1635	1635	1117	52677
9	2390	2390	1365	69166
10	731	731	244	46796
11	2185	1935	1935	142273
12	1387	1387	0	33352
13	809	809	809	21615
14	411	411	411	53012
15	4740	4340	3680	102132
16	1482	1482	0	7094
17	592	592	592	80305
18	1107	1107	1107	99417
19	3863	3213	3213	123923
20	3168	2768	2768	123524
21	1584	1584	1584	100021
22	3042	3042	3042	144796
23	3834	3184	3184	39555
24	4455	4055	4055	190121
25	995	995	663	152933
26	2781	2781	2781	39879
27	1090	1090	209	39999
28	400	400	0	8349
29	2410	2010	1149	44823
...	...	...	...	...
12053	1352	1352	1352	32523
12054	2285	2285	2035	125562
12055	3030	2630	2630	84766
12056	1380	1380	1380	85516
12057	1413	1413	1131	44899
12058	1538	1288	1288	49789
12059	2240	2240	2240	279512
12060	1896	1896	1896	118213
12061	4800	4800	4800	121706
12062	2192	1792	1792	97071
12063	4800	4400	4400	175190

12064	224	224	224	23961
12065	1850	1850	1233	129196
12066	1388	1388	1388	36834
12067	597	597	597	61052
12068	690	690	460	95639
12069	2162	2162	1081	137773
12070	1380	1130	1130	71708
12071	6016	6016	5264	245679
12072	1599	1199	343	153193
12073	3433	3183	1415	101169
12074	1732	1332	1332	77387
12075	2029	2029	2029	93912
12076	748	748	748	28439
12077	3822	3422	3422	73513
12078	2560	2160	893	75702
12079	502	502	502	25251
12080	4581	4181	4181	148252
12081	1728	1728	864	81978
12082	4920	4520	4520	38100

[12083 rows x 11 columns]



```
In [163]: import matplotlib.cm as cm
          import matplotlib.colors as col
          new2 = pd.DataFrame()
          new2['TOTROOMS']=new.TOTROOMS
          new2['HEATROOM']=new.HEATROOM
          new2['WINDOWS']=new.WINDOWS
          new2['VALUE']=new1.VALUE
          fig = plt.figure(figsize=(10,6))
          new2.plot.hexbin(x='TOTROOMS', y='WINDOWS', C='VALUE',reduce_C_function=r

Out[163]: <matplotlib.axes._subplots.AxesSubplot at 0x21d19d296d8>

<matplotlib.figure.Figure at 0x21d19d3c128>
```