Project Overview

Build a predictive model that identifies the month, hour of day, and appliances (kitchen, laundry, or water heater/ac) that use the most power.

Hypothesis

The most power is used by water heater/ac during months of Dec, Jan, Feb, and July between the hours of 6am - 9pm.

Data Overview/Cleansing

- Drop irrelevant rows
- Change column names for easier readability
- Change date column to month only
- Change time column to hour of day (24 hour clock)
- Remove '?' and null values
- Use MinMaxScaler for Kitchen, LaundryRoom, and WaterHeat_AC columns due to variance in values
- Removed zero values due to imbalanced data

Analysis and Results

- Split data into train and test (80%, 20%)
- Target variable > ‘Consumer_active_power?’
- Ran Linear Regression in Scikit Learn and StatsModels
- None of the factors reflected normal distribution
- Very little correlation for any factors
- Scikit Learn provided score of 0.86
- StatsModels provided R-squared of 0.86

In [2]:

```
import pandas as pd
import numpy as np
import csv
import sys
```

In [3]:

```
#df = pd.read_table('household_power_consumption.txt')
with open("household_power_consumption.txt" , "r") as txt_file:
     with open("csv_file.csv", "w") as csv_file:
                in_txt = csv.reader(txt_file, delimiter = ';')
                out_csv = csv.writer(csv_file)
                out_csv.writerows(in_txt)
```

In [4]:

```
df = pd.read_csv('csv_file.csv')
df.head(3)
```

```
/Users/krys/anaconda2/lib/python2.7/site-packages/IPython/core/interac
tiveshell.py:2717: DtypeWarning: Columns (2,3,4,5,6,7) have mixed type
s. Specify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

Out[4]:

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_int
0	16/12/2006	17:24:00	4.216	0.418	234.840	18.400
1	16/12/2006	17:25:00	5.360	0.436	233.630	23.000
2	16/12/2006	17:26:00	5.374	0.498	233.290	23.000

In [5]:

```
df.shape
```

Out[5]:

```
(2075259, 9)
```

In [6]:

```
df.drop(df.columns[[3,4,5]], axis=1, inplace=True)
```

In [7]:

```
df.head(2)
```

Out[7]:

	Date	Time	Global_active_power	Sub_metering_1	Sub_metering_2	Sub_me
0	16/12/2006	17:24:00	4.216	0.000	1.000	17.0
1	16/12/2006	17:25:00	5.360	0.000	1.000	16.0

In [8]:

```
df.rename(columns={'Sub_metering_1':'Kitchen',
                   'Sub_metering_2':'LaundryRoom',
                   'Sub_metering_3':'WaterHeat_AC'},
         inplace=True)
```

In [9]:

```
df.head(2)
```

Out[9]:

	Date	Time	Global_active_power	Kitchen	LaundryRoom	WaterHeat_AC
0	16/12/2006	17:24:00	4.216	0.000	1.000	17.0
1	16/12/2006	17:25:00	5.360	0.000	1.000	16.0

In [10]:

```
from datetime import datetime
from dateutil.parser import parse
```

In [11]:

```
df['Date'] = pd.to_datetime(df['Date'], format='%d/%m/%Y')
df['Date'] = df['Date'].dt.month
```

In [12]:

```
df['Time'] = pd.to_datetime(df['Time'], format='%H:%M:%S')
df['Time'] = df['Time'].dt.hour
```

In [13]:

```
df.head(2)
```

Out[13]:

	Date	Time	Global_active_power	Kitchen	LaundryRoom	WaterHeat_AC
0	12	17	4.216	0.000	1.000	17.0
1	12	17	5.360	0.000	1.000	16.0

In [14]:

```
df['Global_active_power'].value_counts().head()
```

Out[14]:

```
?         25979
0.218      9491
0.216      9319
0.322      9226
0.324      9153
Name: Global_active_power, dtype: int64
```

In [15]:

```
df['Kitchen'].value_counts().head(10)
```

Out[15]:

```
0.000     1840611
1.000       82920
0.0         39564
?           25979
2.000       18537
38.000      15954
37.000      14556
39.000       6452
36.000       5128
1.0          2016
Name: Kitchen, dtype: int64
```

In [16]:

```
df['LaundryRoom'].value_counts().head(10)
```

Out[16]:

```
0.000     1408274
1.000      367317
2.000      153938
0.0         28556
?           25979
1.0         10907
3.000        7096
37.000       6565
4.000        5671
36.000       5498
Name: LaundryRoom, dtype: int64
```

In [17]:

```
df['WaterHeat_AC'].isnull().sum()
```

Out[17]:

```
25979
```

In [18]:

```
df['Global_active_power'].replace('?', np.nan,inplace=True)
```

In [19]:

```
df['Global_active_power'].isnull().value_counts()
```

Out[19]:

```
False    2049280
True       25979
Name: Global_active_power, dtype: int64
```

In [20]:

```
df.dropna(subset=['Global_active_power'], inplace=True)
```

In [21]:

```
df['Global_active_power'].isnull().sum()
```

Out[21]:

```
0
```

In [22]:

```
df['Kitchen'].replace('?', np.nan,inplace=True)
```

In [23]:

```
df['Kitchen'].isnull().value_counts()
```

Out[23]:

```
False    2049280
Name: Kitchen, dtype: int64
```

In [24]:

```
df.dropna(subset=['Kitchen'], inplace=True)
```

In [25]:

```
df['Kitchen'].isnull().sum()
```

Out[25]:

```
0
```

In [26]:

```
#df1 = df1.drop(df1[(df1.LaundryRoom == '?').index])
df['LaundryRoom'].replace('?', np.nan,inplace=True)
```

In [27]:

```
df['LaundryRoom'].isnull().value_counts()
```

Out[27]:

```
False    2049280
Name: LaundryRoom, dtype: int64
```

In [28]:

```
df.dropna(subset=['LaundryRoom'], inplace=True)
```

In [29]:

```
df['LaundryRoom'].isnull().sum()
```

Out[29]:

```
0
```

In [30]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2049280 entries, 0 to 2075258
Data columns (total 6 columns):
Date                      int64
Time                      int64
Global_active_power       object
Kitchen                   object
LaundryRoom               object
WaterHeat_AC              float64
dtypes: float64(1), int64(2), object(3)
memory usage: 109.4+ MB
```

######

In [31]:

```
from sklearn.preprocessing import MinMaxScaler
scale = MinMaxScaler()
```

In [32]:

```
df[['Kitchen']] = scale.fit_transform(df[['Kitchen']].as_matrix())
```

```
/Users/krys/anaconda2/lib/python2.7/site-packages/sklearn/utils/valida
tion.py:429: DataConversionWarning: Data with input dtype object was c
onverted to float64 by MinMaxScaler.
  warnings.warn(msg, _DataConversionWarning)
```

In [33]:

In [33]:

```
df['Kitchen'].value_counts()
```

Out[33]:

```
0.000000    1880175
0.011364      84936
0.022727      19017
0.431818      16119
0.420455      14892
0.443182       6503
0.409091       5270
0.397727       1359
0.454545       1159
0.363636        802
0.159091        702
0.170455        635
0.375000        603
0.147727        600
0.136364        597
0.352273        586
0.193182        580
0.181818        579
0.113636        576
0.204545        563
0.306818        546
0.125000        541
0.386364        534
0.102273        522
0.238636        510
0.034091        507
0.227273        505
0.215909        487
0.340909        484
0.318182        480
              ...
0.613636         53
0.818182         46
0.625000         44
0.806818         41
0.875000         34
0.636364         32
0.886364         32
0.795455         29
0.897727         28
0.681818         26
0.909091         19
0.761364         18
0.772727         16
0.715909         16
0.659091         13
0.750000         13
0.670455         13
0.647727         13
```

```
0.738636        12
0.727273        12
0.704545        12
0.693182        10
0.784091         9
0.920455         6
0.943182         4
0.988636         3
0.931818         3
1.000000         3
0.954545         2
0.977273         2
Name: Kitchen, dtype: int64
```

In [34]:

```
df[['Global_active_power']] = scale.fit_transform(df[['Global_active_power']].as_mat
```

In [35]:

```
df[['LaundryRoom']] = scale.fit_transform(df[['LaundryRoom']].as_matrix())
```

In [36]:

```
df[['WaterHeat_AC']] = scale.fit_transform(df[['WaterHeat_AC']].as_matrix())
```

In [37]:

```
df.head(3)
```

Out[37]:

	Date	Time	Global_active_power	Kitchen	LaundryRoom	WaterHeat_AC
0	12	17	0.374796	0.0	0.0125	0.548387
1	12	17	0.478363	0.0	0.0125	0.516129
2	12	17	0.479631	0.0	0.0250	0.548387

In [38]:

```
df.describe()
```

Out[38]:

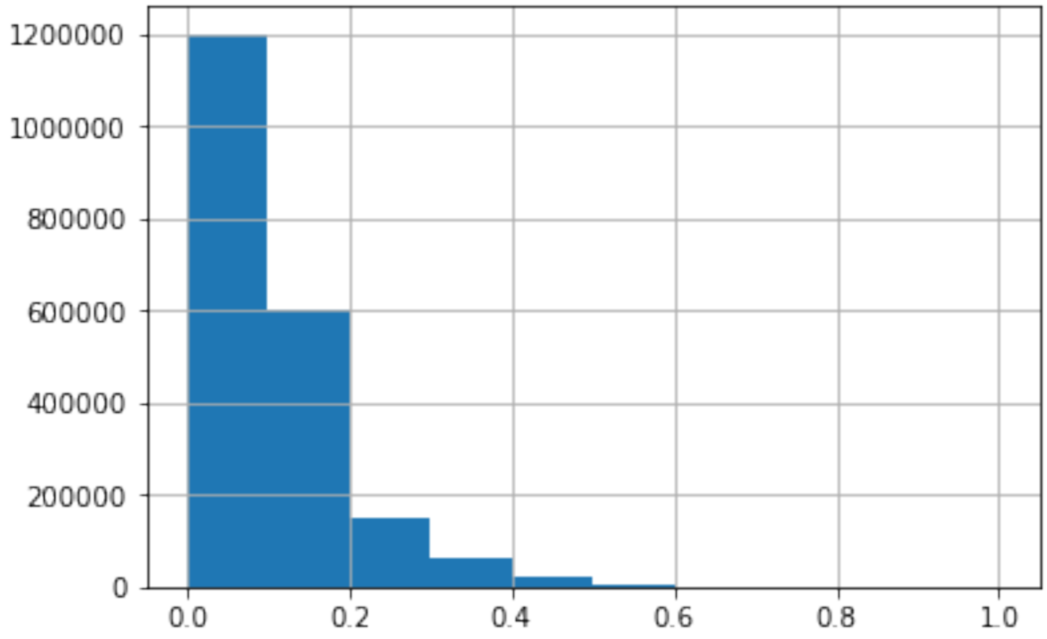
	Date	Time	Global_active_power	Kitchen	LaundryRoom
count	2.049280e+06	2.049280e+06	2.049280e+06	2.049280e+06	2.049280e+06
mean	6.454433e+00	1.150391e+01	9.194415e-02	1.274913e-02	1.623150e-02
std	3.423209e+00	6.925189e+00	9.571738e-02	6.992081e-02	7.277533e-02
min	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	3.000000e+00	5.000000e+00	2.100308e-02	0.000000e+00	0.000000e+00
50%	6.000000e+00	1.200000e+01	4.761905e-02	0.000000e+00	0.000000e+00
75%	9.000000e+00	1.800000e+01	1.314503e-01	0.000000e+00	1.250000e-02
max	1.200000e+01	2.300000e+01	1.000000e+00	1.000000e+00	1.000000e+00

In [39]:

```
import matplotlib.pyplot as plt
%matplotlib inline
df.Global_active_power.hist()
```

Out[39]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1140cc850>
```

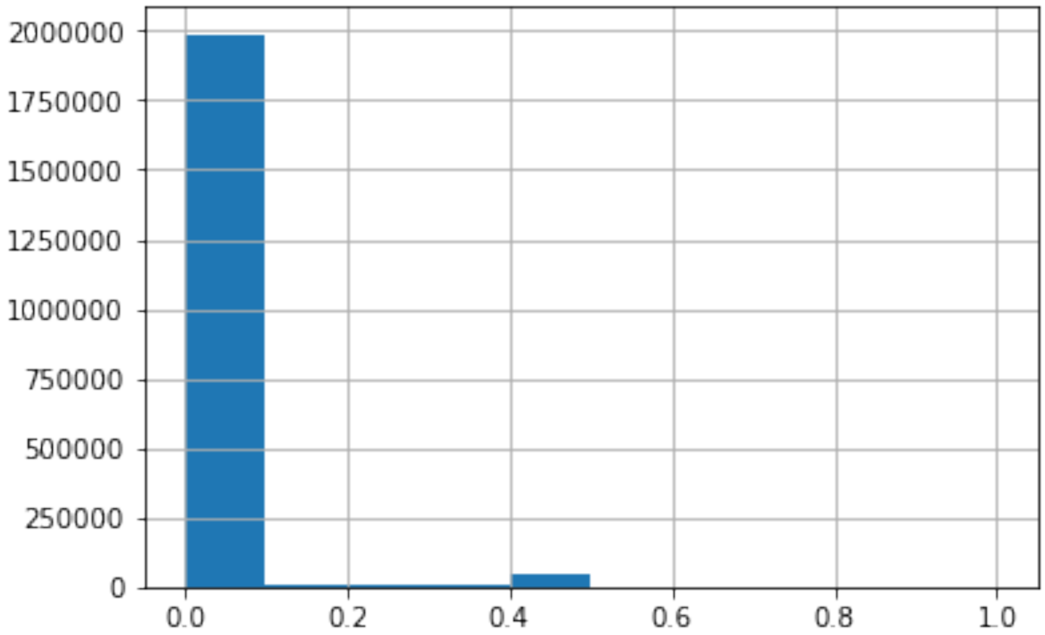


In [40]:

```
df.Kitchen.hist()
```

Out[40]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x11433a610>
```

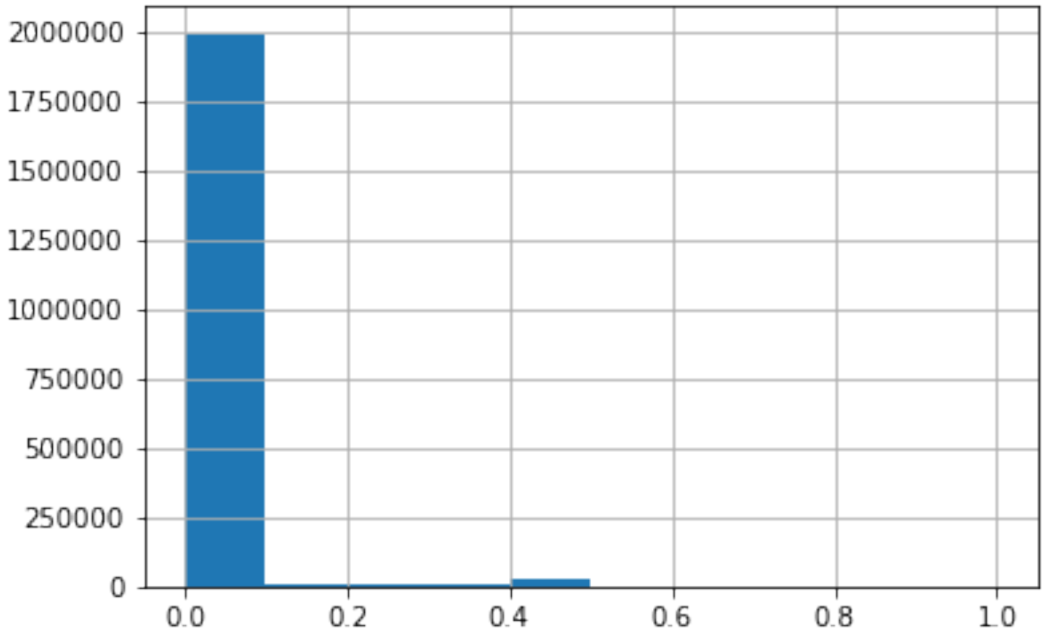


In [41]:

```
df.LaundryRoom.hist()
```

Out[41]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1144e2390>
```

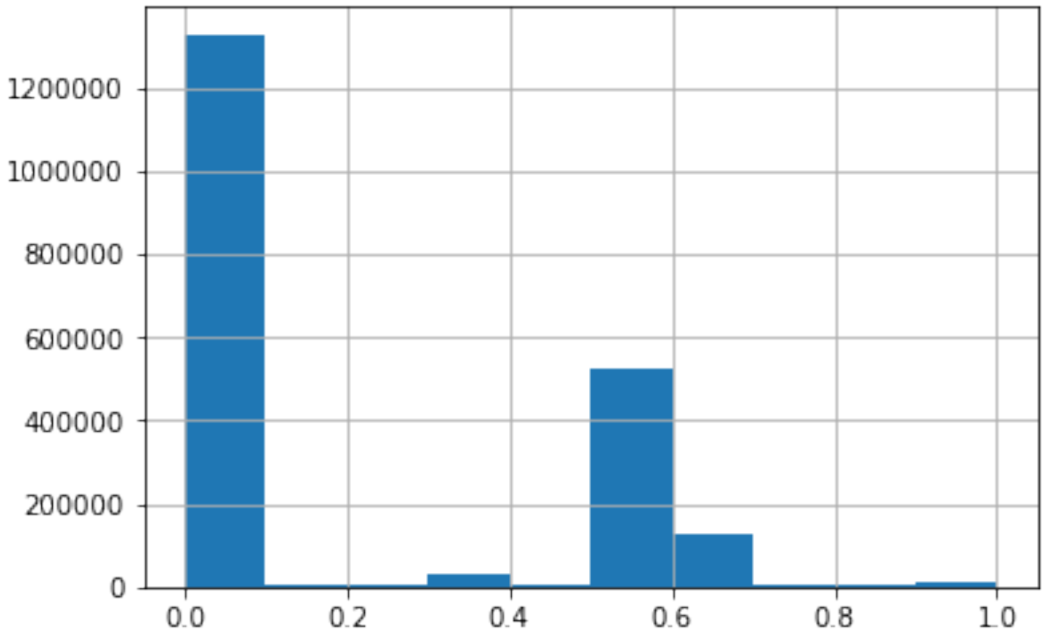


In [42]:

```
df.WaterHeat_AC.hist()
```

Out[42]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x114669c90>
```



In [46]:

```
print df.groupby('Date')['Kitchen']
```

```
<pandas.core.groupby.SeriesGroupBy object at 0x114a4c590>
```

In [47]:

```
df = df[df.Kitchen != 0.0]
```

In [48]:

```
df = df[df.LaundryRoom != 0.0]
```

In [49]:

```
df = df[df.WaterHeat_AC != 0.0]
```

In [50]:

```
df['Kitchen'].value_counts().head()
```

Out[50]:

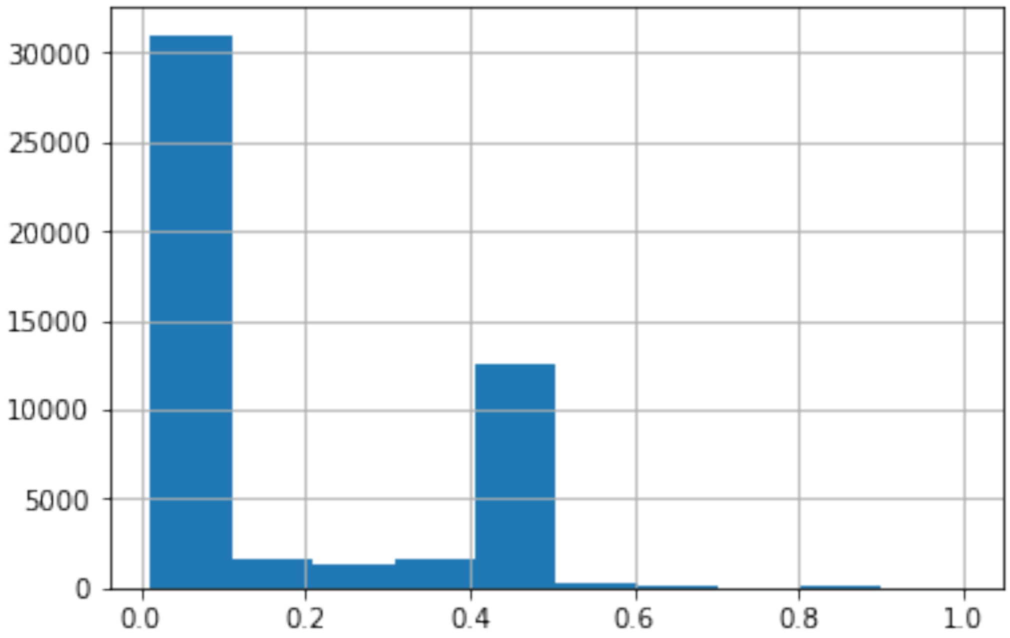
```
0.011364    24533
0.022727     5562
0.420455     4523
0.431818     4188
0.409091     1823
Name: Kitchen, dtype: int64
```

In [51]:

```
df.Kitchen.hist()
```

Out[51]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x114a4cd50>
```

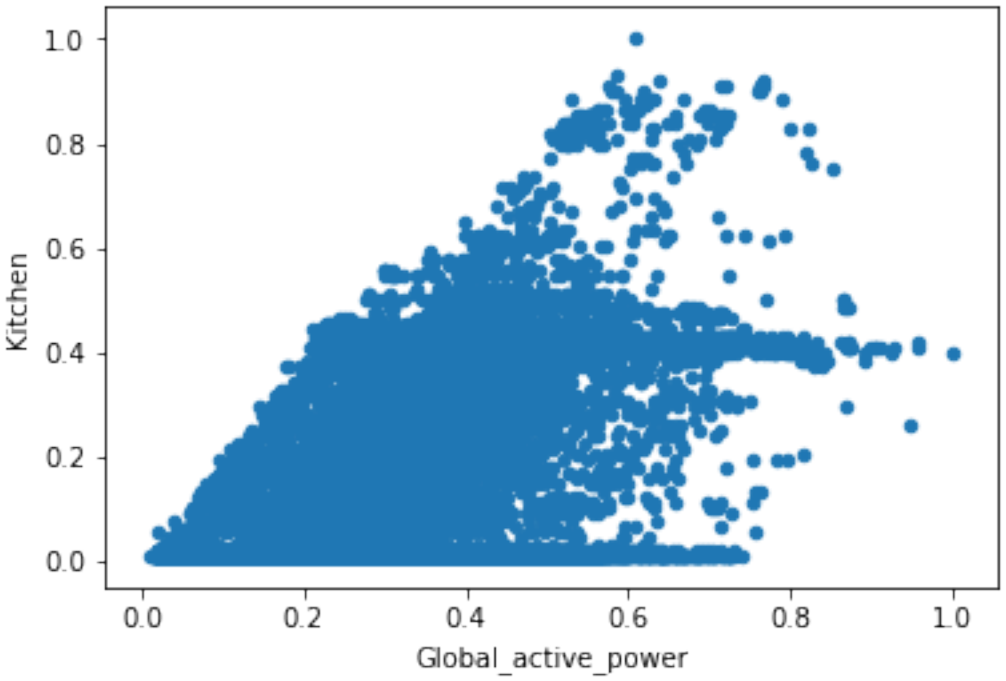


In [52]:

```
df.plot(x='Global_active_power', y='Kitchen', kind='scatter')
```

Out[52]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x114aa0250>
```

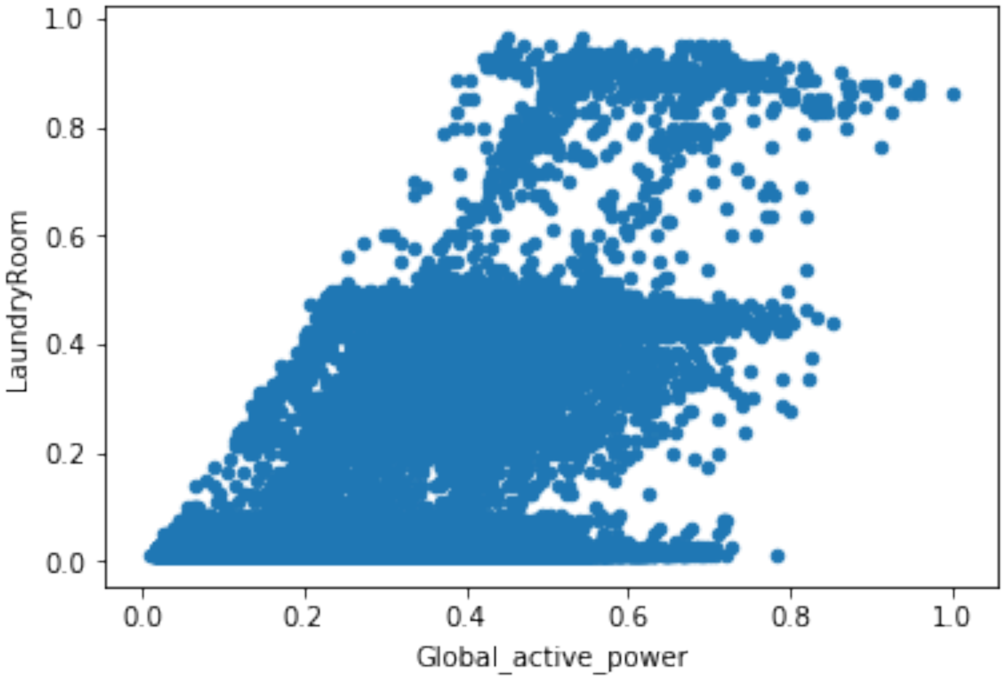


In [53]:

```
df.plot(x='Global_active_power', y='LaundryRoom', kind='scatter')
```

Out[53]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x11706fe90>
```

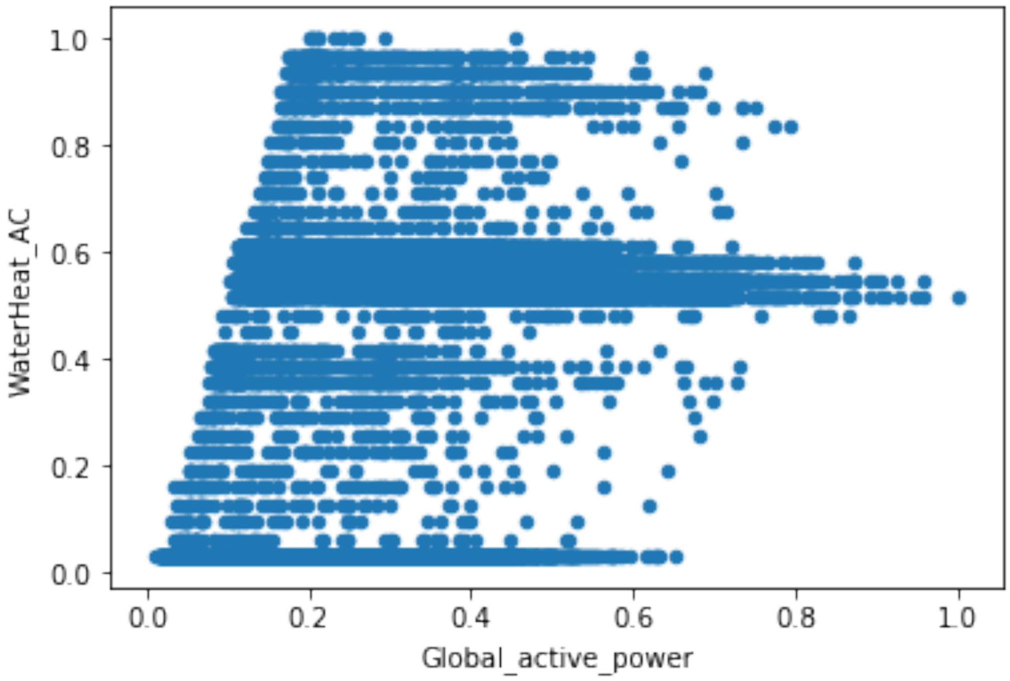


In [54]:

```
df.plot(x='Global_active_power', y='WaterHeat_AC', kind='scatter')
```

Out[54]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1168c9450>
```

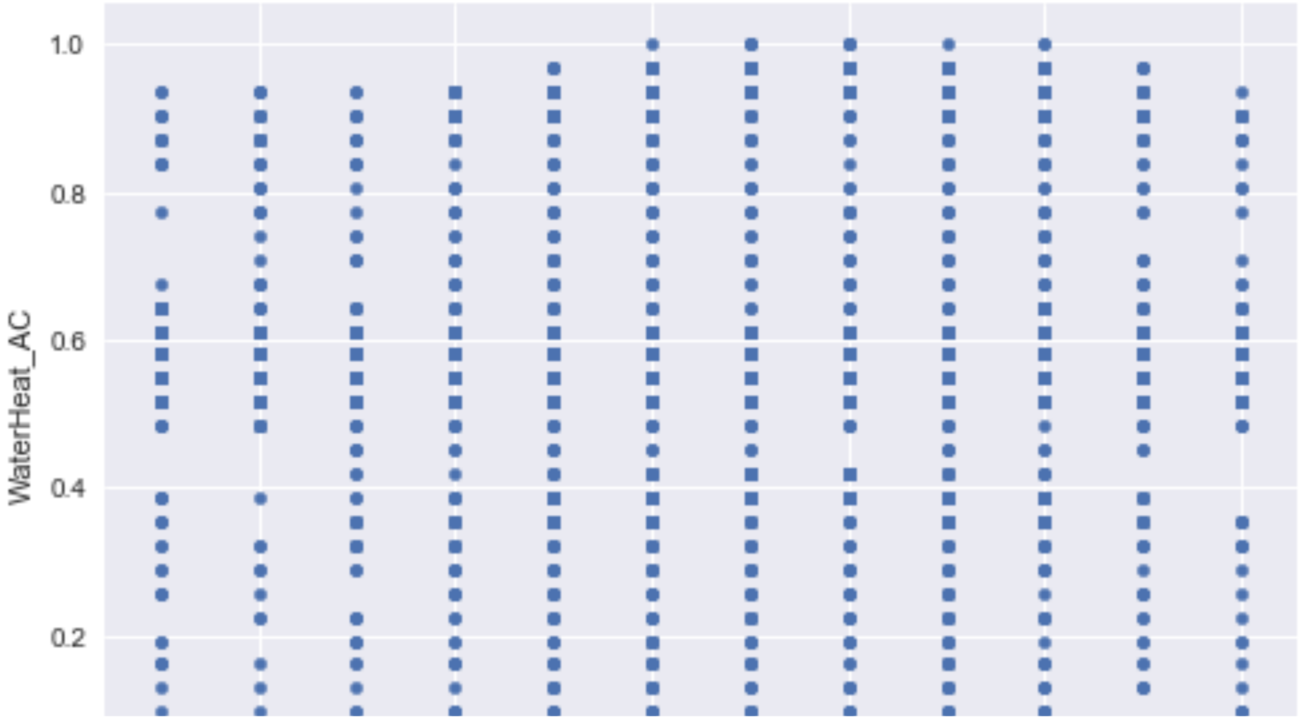


In [92]:

```
df.plot(x='Date', y='WaterHeat_AC', kind='scatter')
```

Out[92]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x11995f190>
```

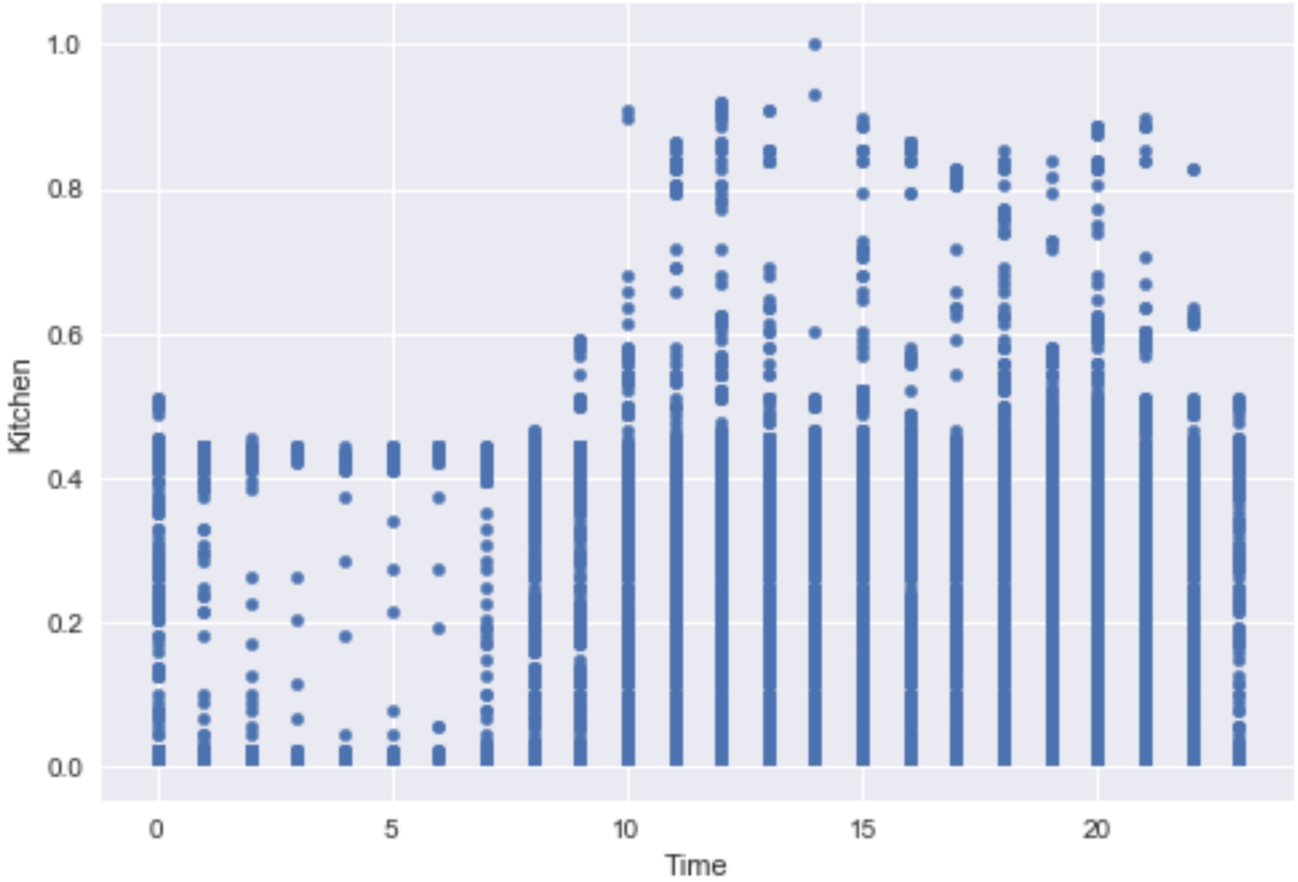


In [94]:

```
df.plot(x='Time', y='Kitchen', kind='scatter')
```

Out[94]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1199eb910>
```

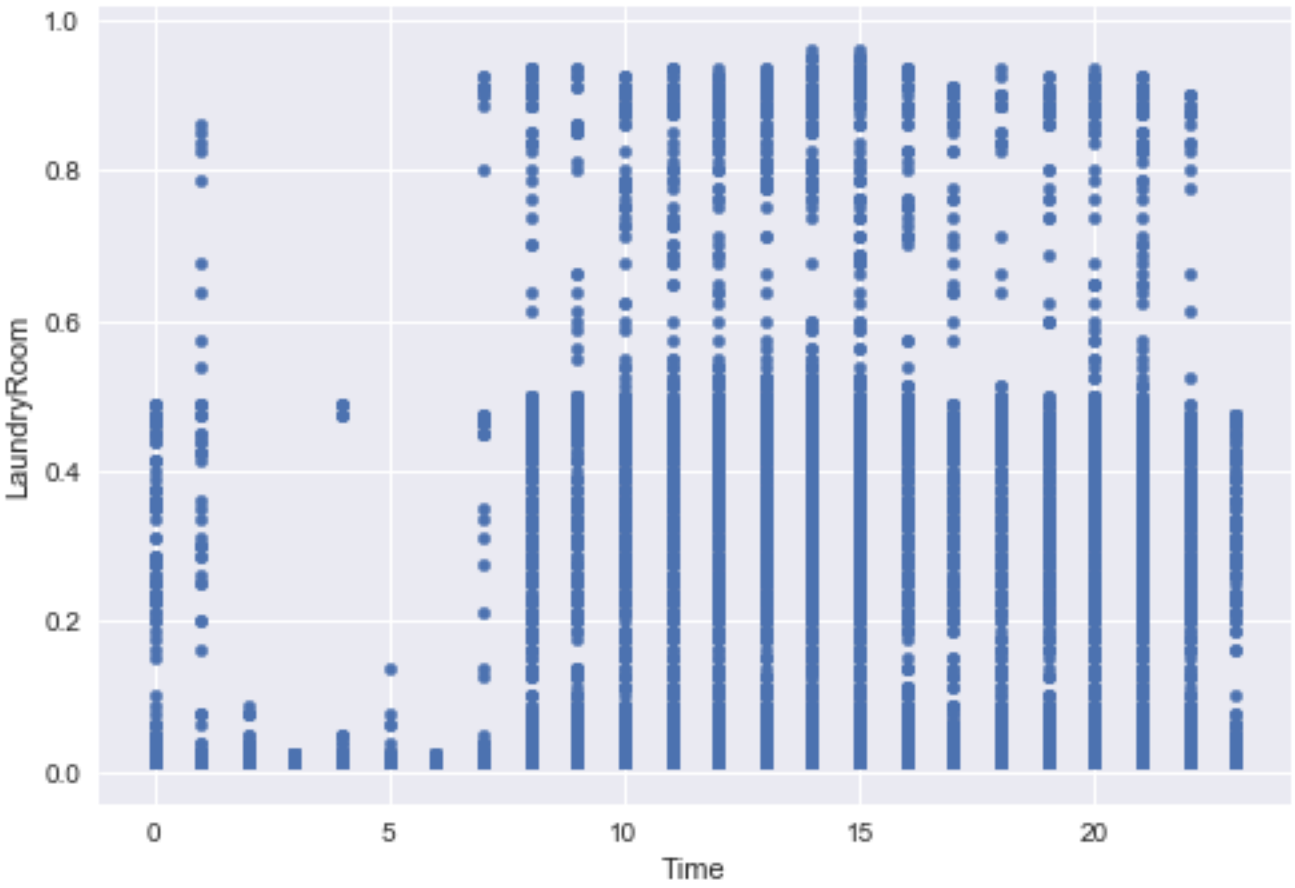


```
In [95]:
```

```
df.plot(x='Time', y='LaundryRoom', kind='scatter')
```

```
Out[95]:

<matplotlib.axes._subplots.AxesSubplot at 0x11a05b150>
```



```
In [55]:
```

```
df.groupby('Global_active_power')['Kitchen', 'LaundryRoom', 'WaterHeat_AC'].head()
```

```
Out[55]:
```

	Kitchen	LaundryRoom	WaterHeat_AC
1019	0.011364	0.0875	0.548387
1020	0.022727	0.4375	0.548387
1021	0.011364	0.3250	0.548387
1022	0.011364	0.4375	0.548387
1023	0.011364	0.3375	0.548387
1024	0.011364	0.4375	0.548387
1025	0.022727	0.4375	0.548387
1026	0.011364	0.4375	0.548387
1027	0.011364	0.4625	0.548387
1028	0.011364	0.4500	0.548387

In [56]:

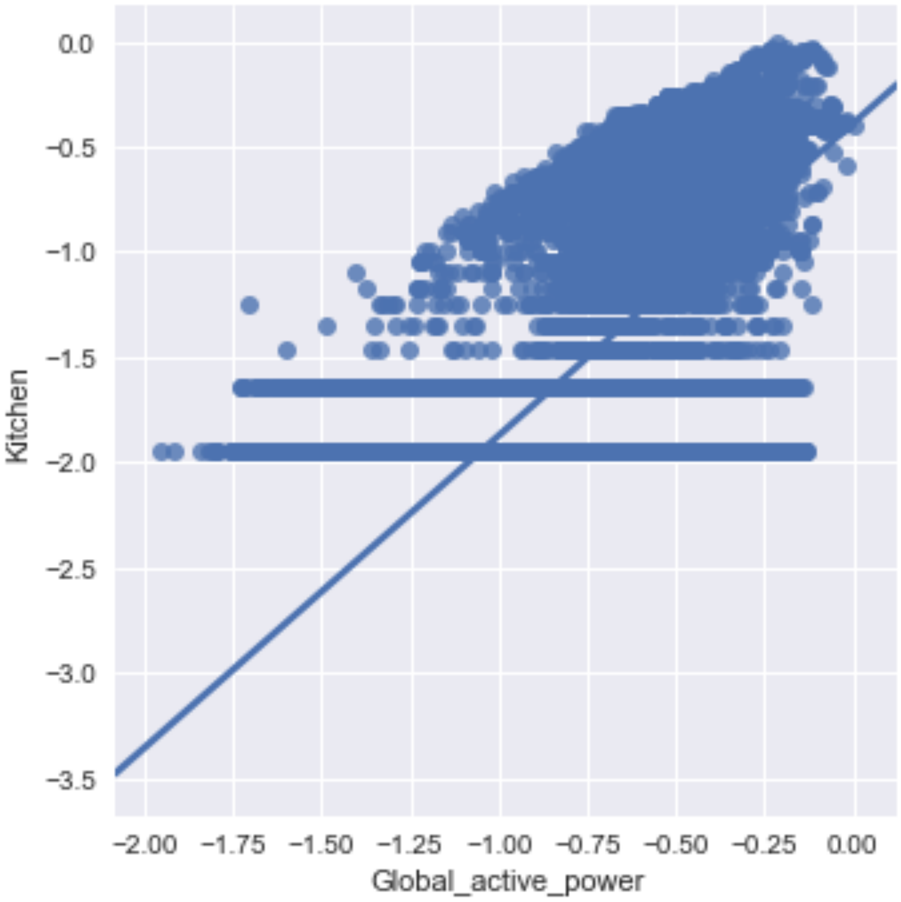
```
log_columns = ['Global_active_power', 'Kitchen']
log_df = df.copy()
log_df[log_columns] = log_df[log_columns].apply(np.log10)
```

In [57]:

```
import seaborn as sns
sns.lmplot('Global_active_power', 'Kitchen', log_df)
```

Out[57]:

```
<seaborn.axisgrid.FacetGrid at 0x116a1a450>
```



In [58]:

```
df.corr()
```

Out[58]:

	Date	Time	Global_active_power	Kitchen	LaundryRoo
Date	1.000000	-0.029764	-0.059210	-0.012303	-0.030546
Time	-0.029764	1.000000	0.124053	-0.000787	-0.029223
Global_active_power	-0.059210	0.124053	1.000000	0.645848	0.586256
Kitchen	-0.012303	-0.000787	0.645848	1.000000	0.006576
LaundryRoom	-0.030546	-0.029223	0.586256	0.006576	1.000000
WaterHeat_AC	-0.085688	-0.122444	0.305524	0.009362	0.056804

In [59]:

```
#from sklearn.model_selection import StratifiedKFold
#x_train, x_test, y_train, y_test = cross_validation.train_test_split(x,y,
#                                                                    test_size=0.20,
 #                                                                   random_state=0,
```

In [60]:

```
df = df[['Date', 'Time', 'Kitchen', 'LaundryRoom', 'WaterHeat_AC', 'Global_active_po
```

In [61]:

```
df.head()
```

Out[61]:

	Date	Time	Kitchen	LaundryRoom	WaterHeat_AC	Global_active_power
1019	12	10	0.011364	0.0875	0.548387	0.196089
1020	12	10	0.022727	0.4375	0.548387	0.329350
1021	12	10	0.011364	0.3250	0.548387	0.283904
1022	12	10	0.011364	0.4375	0.548387	0.327539
1023	12	10	0.011364	0.3375	0.548387	0.283179

In [62]:

```
from sklearn.cross_validation import train_test_split
```

```
/Users/krys/anaconda2/lib/python2.7/site-packages/sklearn/cross_valida
tion.py:44: DeprecationWarning: This module was deprecated in version
0.18 in favor of the model_selection module into which all the refacto
red classes and functions are moved. Also note that the interface of t
he new CV iterators are different from that of this module. This modul
e will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
```

In [63]:

```
train, test = train_test_split(df, train_size=.80, test_size=.20)
```

In [64]:

```
from sklearn import linear_model
from sklearn.linear_model import LinearRegression
lin_model = linear_model.LinearRegression()
```

In [80]:

```
feature_cols = ['Date', 'Time', 'Kitchen', 'LaundryRoom', 'WaterHeat_AC']
X = train[feature_cols]
y = train.Global_active_power
```

In [81]:

```
lin_model.fit(X,y)
```

Out[81]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=
False)
```

In [86]:

```
print lin_model.intercept_
print lin_model.coef_
```

```
-0.0232587114105
[ -1.17121708e-04   4.97507939e-03   4.93865726e-01   4.63246609e-01
   1.97212089e-01]
```

In [84]:

```
pd.DataFrame(zip(train.columns, lin_model.coef_), columns = ['factors', 'est_Coef']
```

Out[84]:

	factors	est_Coef
0	Date	-0.000117
1	Time	0.004975
2	Kitchen	0.493866
3	LaundryRoom	0.463247
4	WaterHeat_AC	0.197212

In [88]:

```
lin_model.score(X,y)
```

Out[88]:

```
0.85724536127314077
```

In [89]:

```
test_feature_cols = ['Date', 'Time', 'Kitchen', 'LaundryRoom', 'WaterHeat_AC']
test_X = test[feature_cols]
test_y = test.Global_active_power
```

In [91]:

```
lin_model.score(test_X,test_y)
```

Out[91]:

```
0.86002406046229396
```

In [70]:

```
# STATS MODELS
```

In [71]:

```
import statsmodels.formula.api as smf
```

In [72]:

```
lm = smf.ols(formula='Global_active_power ~ Kitchen + LaundryRoom + WaterHeat_AC',
             data=df).fit()
```

In [73]:

```
lm.params
```

Out[73]:

```
Intercept       0.059622
Kitchen         0.493899
LaundryRoom     0.459931
WaterHeat_AC    0.183372
dtype: float64
```

In [75]:

```
lm.summary()
```

Out[75]:

OLS Regression Results

Dep. Variable:	Global_active_power	**R-squared:**	0.827
Model:	OLS	**Adj. R-squared:**	0.827
Method:	Least Squares	**F-statistic:**	7.764e+04
Date:	Mon, 20 Feb 2017	**Prob (F-statistic):**	0.00
Time:	18:02:46	**Log-Likelihood:**	66873.
No. Observations:	48693	**AIC:**	-1.337e+05
Df Residuals:	48689	**BIC:**	-1.337e+05
Df Model:	3		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[95.0% Conf. Int.]**
Intercept	0.0596	0.001	82.304	0.000	0.058 0.061
Kitchen	0.4939	0.001	339.401	0.000	0.491 0.497
LaundryRoom	0.4599	0.002	300.323	0.000	0.457 0.463
WaterHeat_AC	0.1834	0.001	141.631	0.000	0.181 0.186

Omnibus:	10314.988	**Durbin-Watson:**	0.184
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	19654.854
Skew:	1.310	**Prob(JB):**	0.00
Kurtosis:	4.682	**Cond. No.**	6.26

In [76]:

```
lm = smf.ols(formula='Global_active_power ~ Kitchen + LaundryRoom + WaterHeat_AC + I
             data=df).fit()
```

In [77]:

```
lm.summary()
```

Out[77]:

OLS Regression Results

Dep. Variable:	Global_active_power	**R-squared:**	0.858
Model:	OLS	**Adj. R-squared:**	0.858
Method:	Least Squares	**F-statistic:**	5.875e+04
Date:	Mon, 20 Feb 2017	**Prob (F-statistic):**	0.00
Time:	18:09:10	**Log-Likelihood:**	71635.
No. Observations:	48693	**AIC:**	-1.433e+05
Df Residuals:	48687	**BIC:**	-1.432e+05
Df Model:	5		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[95.0% Conf. Int.]**
Intercept	-0.0225	0.001	-18.926	0.000	-0.025 -0.020
Kitchen	0.4938	0.001	374.162	0.000	0.491 0.496
LaundryRoom	0.4630	0.001	333.212	0.000	0.460 0.466
WaterHeat_AC	0.1978	0.001	166.581	0.000	0.195 0.200
Date	-0.0002	7.49e-05	-2.299	0.022	-0.000 -2.54e-05
Time	0.0049	4.82e-05	102.350	0.000	0.005 0.005

Omnibus:	12342.380	**Durbin-Watson:**	0.216
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	33023.272
Skew:	1.363	**Prob(JB):**	0.00
Kurtosis:	5.974	**Cond. No.**	105.

In [ ]:

In [ ]:

In [ ]: