Electricity Price Prediction for Chicago Area

Arpit Garg

16th November, 2017

1 Introduction

ComEd, an Exelon Company provides electricity and is the largest electric utility in Illinois, serving Chicago and Northern Illinois. They have APIs[1] for 5-minute prices for ComEd's Hourly Pricing Program. Different data that can be found using these APIs are, returns all 5-minute prices from the last 24 hours, returns 5-minute prices between the times provided, inclusively and hour average prices. We can use this data to predict future prices using regression models.

Problem Solving Framework:

- Business Issue Understanding
- Data understanding and Data Preparation(fetching from API)
- Analysis/ Modeling
- Validation
- Visualization/ Presentation.

We will be using Linear Regression and ARIMA models to predict the prices of future electricity usage. This will help in predicting future requirements. We can build the models using Python packages and then show the results graphically.

2 Work Done

Right now the business issue understanding and data understanding and preparation has been done and the work on the analysis/modeling part is going on. The data has been fetched from the API and stored in Pandas DataFrame for better analysis.

[1] https://hourlypricing.comed.com/hp-api/

The following screen shot shows the data of the electricity prices fetched from 1st to 15th November.

In [84]:
```
def get_prices(startDate,endDate=None):
    '''
    gets data from ComEd
    startDate and endDate should be in YYYYMMDDhhmm
    output is a list of {"1234",5.5}, where "1234" is the UTC in seconds and 5.5 is the price
    '''
    endDate = startDate if endDate is None else endDate
    startDate=Chicago.normalize(Chicago.localize(dateutil.parser.parse(startDate)))
    endDate=Chicago.normalize(Chicago.localize(dateutil.parser.parse(endDate)))
    format="%Y%m%d%H%M"
    startDate=startDate.strftime(format)
    endDate=endDate.strftime(format)
    URL="https://hourlypricing.comed.com/api?type=5minutefeed&datestart=STARTDATE&dateend=ENDDATE&format=json"
    temp=URL.replace("STARTDATE",startDate).replace("ENDDATE",endDate)
    print(temp)
    r = requests.get(url=temp)
    data = r.json()
#     print [int(obj.values()[1])/1000 for obj in data]
    out={}
    for obj in data:
        try:
            ts=int(obj.values()[1])/1000
            dt=Chicago.normalize(Chicago.localize(datetime.datetime.fromtimestamp(ts)))
            out[dt]=float(obj.values()[0])
        except Exception:
            pass
#     print out
    return out
```

In [90]:
```
startDate="20171101"
endDate="20171115"

data=get_prices(startDate,endDate)
print(str(len(data.keys()))+" records of data")
```

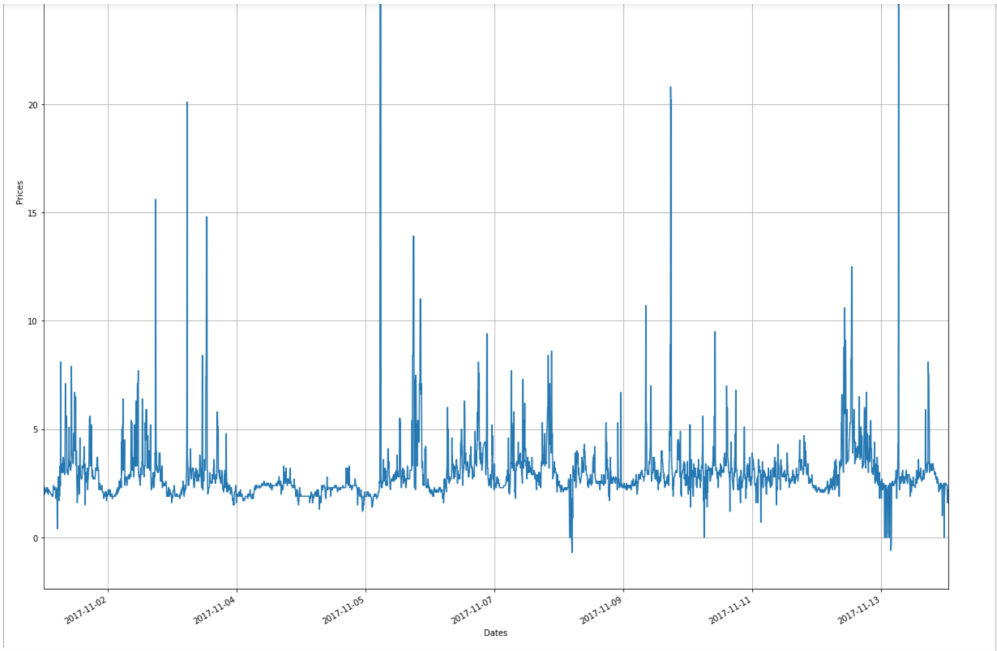
```
https://hourlypricing.comed.com/api?type=5minutefeed&datestart=201711010000&dateend=201711150000&format=json
3947 records of data
```

After that the data is stored in Pandas DataFrame and it can be seen here.

Out[197]:

2017-11-01 00:00:00-05:00	2.4
2017-11-01 00:05:00-05:00	2.0
2017-11-01 00:10:00-05:00	2.2
2017-11-01 00:15:00-05:00	2.1
2017-11-01 00:20:00-05:00	2.0
2017-11-01 00:25:00-05:00	2.3
2017-11-01 00:30:00-05:00	2.1
2017-11-01 00:35:00-05:00	2.1
2017-11-01 00:40:00-05:00	2.2
2017-11-01 00:45:00-05:00	2.2
2017-11-01 00:50:00-05:00	2.2
2017-11-01 00:55:00-05:00	2.2
2017-11-01 01:00:00-05:00	2.1
2017-11-01 01:05:00-05:00	2.2
2017-11-01 01:10:00-05:00	2.3
2017-11-01 01:15:00-05:00	2.3
2017-11-01 01:20:00-05:00	2.1
2017-11-01 01:25:00-05:00	2.3
2017-11-01 01:30:00-05:00	2.2
2017-11-01 01:35:00-05:00	2.0
2017-11-01 01:40:00-05:00	2.0
2017-11-01 01:45:00-05:00	2.0
2017-11-01 01:50:00-05:00	2.2
2017-11-01 01:55:00-05:00	2.2
2017-11-01 03:10:00-05:00	1.9
2017-11-01 03:15:00-05:00	2.0
2017-11-01 03:20:00-05:00	2.0
2017-11-01 03:25:00-05:00	1.9
2017-11-01 03:30:00-05:00	2.0
2017-11-01 03:35:00-05:00	2.3
...	...
2017-11-14 21:05:00-06:00	2.5
2017-11-14 21:10:00-06:00	2.3
2017-11-14 21:15:00-06:00	2.3
2017-11-14 21:20:00-06:00	2.5
2017-11-14 21:25:00-06:00	2.5
2017-11-14 21:30:00-06:00	2.4
2017-11-14 21:35:00-06:00	2.4
2017-11-14 21:40:00-06:00	2.1
2017-11-14 21:45:00-06:00	1.0

Here we can see the plot of the prices for 15 days at an interval of 15 minutes.



After this we will perform regression analysis such as linear regression and time-series regression analysis such as AR, MA and ARIMA for finding the best predictive model. Using cross validation we can test the accuracy of the data by splitting the dataset into 75%(training) and 25%(testing). Then from the model from which we get the best accuracy, we will use that predict the prices for the next day. We can then plot that and create a dashboard for predicted values.

One of the main challenges is to identify the best possible solution for such a prediction so we will try to compare few of these methods. Also since there are not much feature variables, the results will be based on a cluster analysis.

Customer - Akash Narayan, Engineer at Mortenson, Chicago

Email: akash.narayan@mortenson.com

I selected this customer because he lives in Chicago working full time so he knows about the price he is paying for electricity and it can be helpful for such a customer to analyze the prices and can help them reduce energy costs.