12-752: Data-Driven Energy Management of Buildings Assignment#2

Mario Bergés
Assistant Professor
Department of Civil and Environmental Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
`marioberges@cmu.edu`

November 14, 2014

Some notes before you begin:

- Make sure you document everything you do, and not just write down the answer to the question. This will both help during grading as well as improving your learning process.
- Do not write down any solution or process that you do not understand. If you feel that you do not understand how to do something, seek some help: e-mail the TAs or the instructor, post a comment on the BlackBoard discussion board, etc.
- To submit your assignment, please do so using Blackboard. Two files should be uploaded: (a) the Python or MATLAB scripts (i.e. a .py or a .m file); and (b) a PDF file which shows the snippets of code you wrote along with the generated output (including graphs) for each task.
- If you used the IPython Notebook, you should also include the .ipynb file in your submission.
- Please upload a single compressed ZIP file containing the above, and name it as follows: *andrewID_assignment_#.zip* (where *andrewID* is your AndrewID and # is the assignment number.

1 Preface

In our last assignment, we started by providing some guidelines for how to configure your development environment for Python. You will need a similar environment to solve this assignment, though if you have not yet properly configured an environment to your liking, here is one additional suggestion:

- John Kitchin's Blog Post Listing Python Distributions: This blog post has a section about easy-to-install python distributions. You can try some of these to get your environment configured more quickly.

1.1 Loading the Dataset

In this assignment we will be utilizing a dataset of historical electric power measurements collected for a number of buildings on campus during the last 12 months. The dataset can be found on Blackboard or, alternatively, there is a copy in the portoalegre.andrew.cmu.edu server, under the folder /usr/share/12-752/data/.

Just as with the last assignment, the first step will be to laod the CSV file into memory. There are many ways of doing this, but I found the genfromtxt function provided by **numpy** to be quite useful for this task.

Also, because we have one column that contains timestamps, and we want to make use of this information appropriately, you will need to make sure that you convert the timestamp strings (e.g., *2014/09/12 10:11:12*) into a representation that you can later perform numerical operations on. I recommend that you use the `datetime` module. Specifically, the `strptime` method that is provided by it.

Task #1 [10%]: Load the CSV file into memory. Specifically, store the contents in a variable named `data`.

Note: although there are hundreds of ways of doing this, I suggest that you load the data into a `numpy.ndarray` of 744,892 tuples. Each tuple will contain three elements (corresponding to the first three columns of the CSV file) with the following three `dtype`s: ('a255', 'datetime64', float). The first `dtype` corresponds to a string containing at most 255 characters (more info on this), the second one is the data type used by **numpy** to represent dates and times (more info), and the last one corresponds to a floating point number. Another useful date/time data type you could use is Python's built-in type from the `datetime` module.

In the next few sections, we again relax the constraints on how to implement the solutions using Python (or any other programming environment), and simply focus on answering questions using the dataset we have loaded into memory.

2 Exploring the Dataset

In this section we will, again, explore the dataset using common Exploratory Data Analysis techniques.

Task #2 [5%]: How many unique 'Point_name' values are there?

Each Point_name corresponds to a specific power meter (or combination of power meters) for which electrical power measurements are reported.

Task #3 [5%]: What are the names of these power meters (i.e. what are these unique values)?

Task #4 [5%]: How many samples from each power meter are in the dataset?

Task #5 [5%]: For each power meter, how many days are there between the first reported sample and the last?

Task #6 [5%]: Are there any gaps in the reported measurements? If so, where are they?

Let's assume that each power meter should report measurements every n seconds. A gap is defined as a period of time between two consecutive measurements, which exceeds n. In other words, if the measurements $x_1, x_2, x_3, ..., x_m$ are taken at time $t_1, t_2, t_3, ..., t_m$, respectively, then a gap is any pair (t_{i-1}, t_i) for which $t_i - t_{i-1} > n$. Moreover, since we don't really know n for every meter ahead of time, we can estimate it by calculating the mode of the time difference between every two consecutive samples (i.e., the mode of all pairs (t_{i-1}, t_i) $\forall i \in [1, m]$.

Hint: it may be useful to plot the timestamps to quickly visualize any discontinuities.

> **Task #7 [10%]:** Now you will separate the dataset into seven separate variables, each one containing only the measurements that fall within each day of the week. In other words, you will have a variable named `monday`, which only contains the rows for those measurements whose timestamp corresponds to a Monday, another variable by the name of `tuesday`, and so on.

Hint: it may be useful to leverage the method `weekday()` provided by the datetime module.

> **Task #8 [15%]:** For each of the meters in the dataset (7 total), generate a figure for every day of the week (7 total), and in each of those 49 figures generate a plot of all of the corresponding values from the dataset (i.e., the values that belong to each day/meter). These plots should have time on the horizontal axis (a 24 hour period), and power on the vertical axis. Note that the exact date does not matter anymore, so all Mondays will be plotted on top of each other, as well as all Tuesdays, etc.

Hint: you will need to familiarze yourself with subplots, figures and loops if you want to do this efficiently. You will also want to manipulate the dates such that they all fall within the same 24-hour period.

> **Task #9 [10%]:** Explain, in your own words, what you have learned from looking at the data in this way. Feel free to use addtional quantitative explorations here (i.e. calculate the variacne and mean of each day of the week, compare the minimum and maximum values directly, etc.).
> What are the implications of these findings if you want to obtain a linear model that can simulate the power demand of each one of these buildings, say, using linear regression?

3 Linear Regression

Now that the data is ready for us, we will have a little fun with linear regression.

3.1 A case of Mondays...

> **Task #10 [15%]:** Let's focus on Mondays for the meter named "Electric kW Calculations - Main Campus kW".
> Using all the data available (i.e. the same data that you plotted earlier for this meter and day of the week), fit a 3rd order polynomial. Then, using these coefficients, calculate the R^2 value. Finally, separate each of the 52 Mondays that are contained in the dataset for this particular meter, and calculate the R^2 value of your model on each one of those 52 Mondays.

> **Task #11 [5%]:** Which Monday, from the previous task, gave you the worst R^2 value? Why? What can you learn from this?

Task #12 [10%]: In Task #10, the model you were trying to fit was of the form $p(t) = a_0t^3 + a_1t^2 + a_2t + a_3$, where p is the power and t is the time. Now, let's repeat the process but using the following model, which is also linear with respect to the parameters: $p(t) = a_0 + \sum_{k=1}^{5} a_k cos(kt) + b_k sin(kt)$.

Note: for this task you will not be able to use `polyfit`, and should resort to solving the problem using matrix representation as we saw in class.