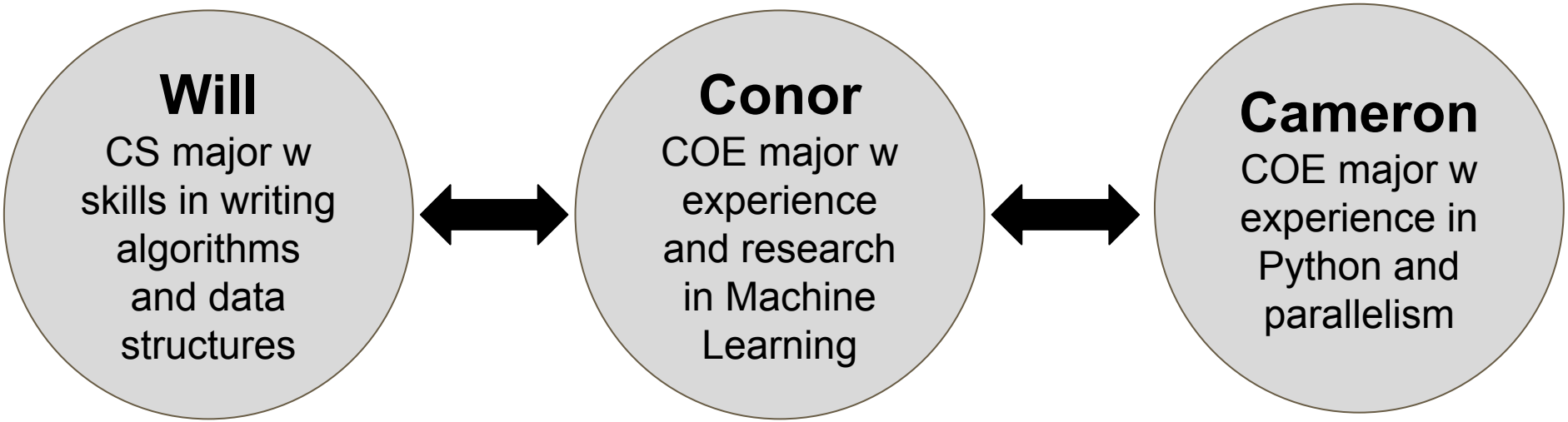
Team Cup Ramen

Cameron Cummins
Conor Donihoo
Will Chin

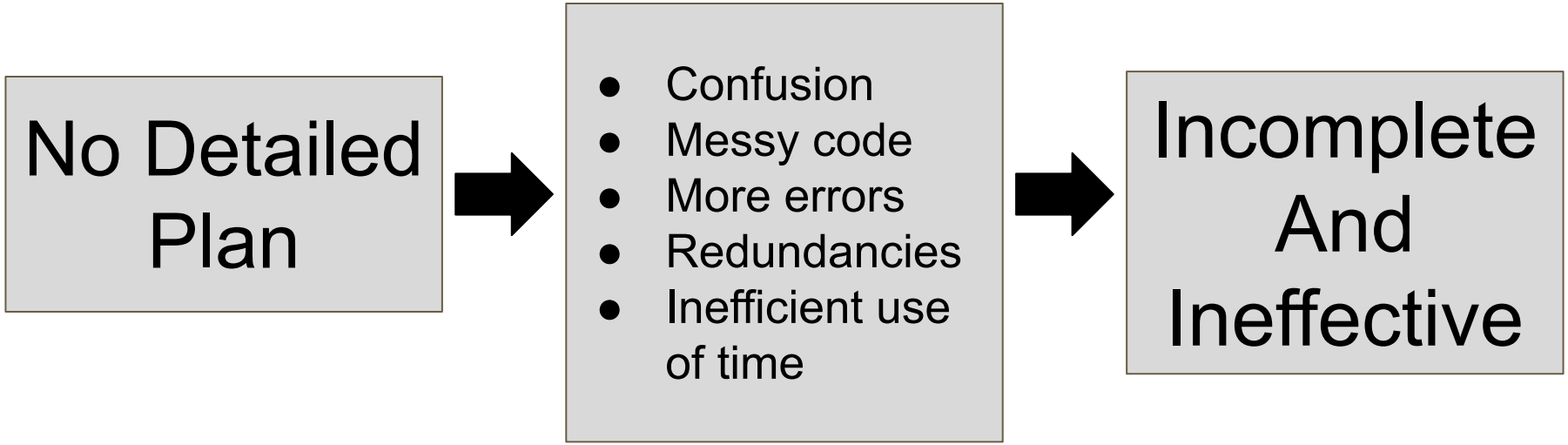
Some Background on Our Team

- This was our first hackathon, so we didn't know what to expect
- Our knowledge was initially divided, so we had to teach each other in order to progress through the hackathon



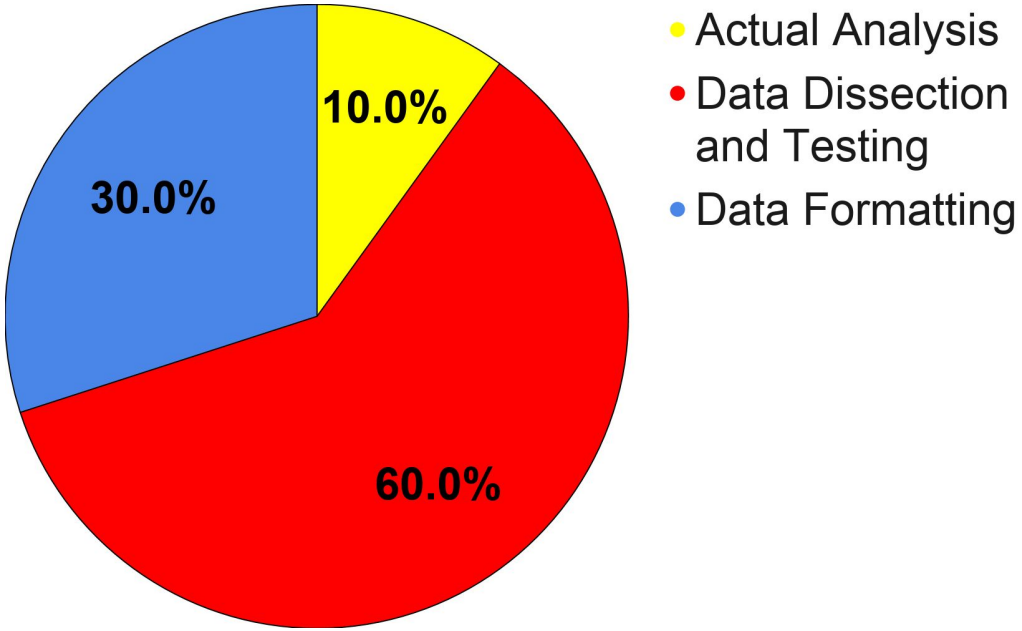
We Made Some Mistakes...

- We made a lot of mistakes in the beginning and we didn't form a well-thought-out *plan*



We Made Some Mistakes...

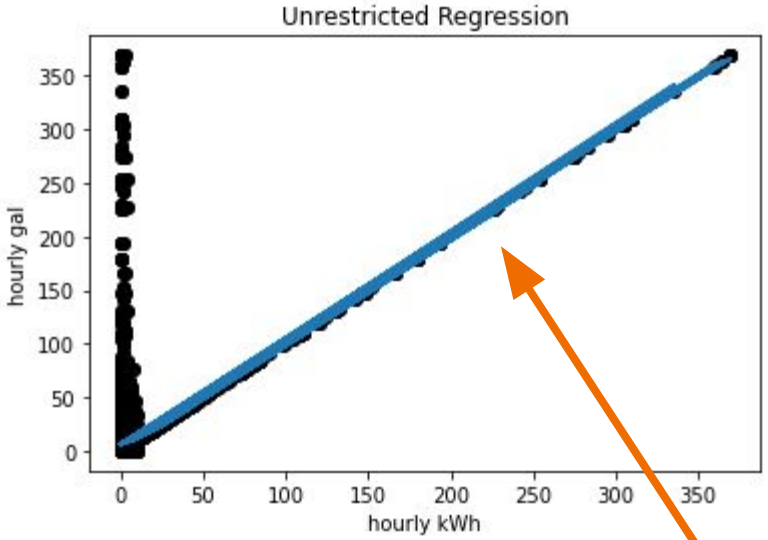
- We ultimately spent more time on understanding the data and formatting it into something we thought we could use than actually analyzing it

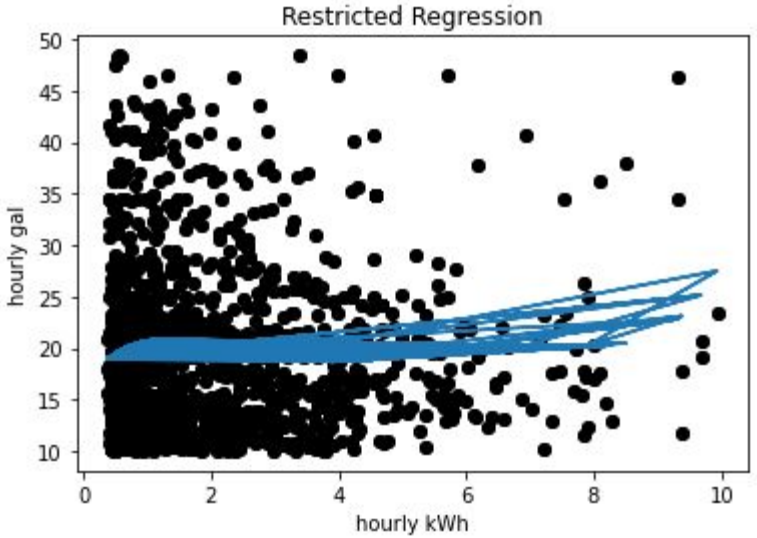


Specific Mistakes

- Codependency and reliance
- Working past efficient hours
- Not focusing on our end-goal
- Misunderstanding the data

Problems we ran into:





O(n)^2
INEFFICIENT!

Custom database
bug that persisted

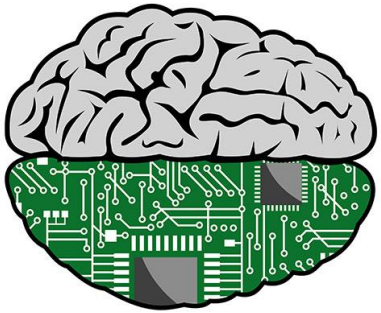
sortWeather.py Added failures 8 minutes ago

Other Problems we ran into:

- Finding good documentation for a regressor
- Translating matlab to python
- Originally formatting data by hIDs
- Formatting data by time
- Running out of time

Machine Learning

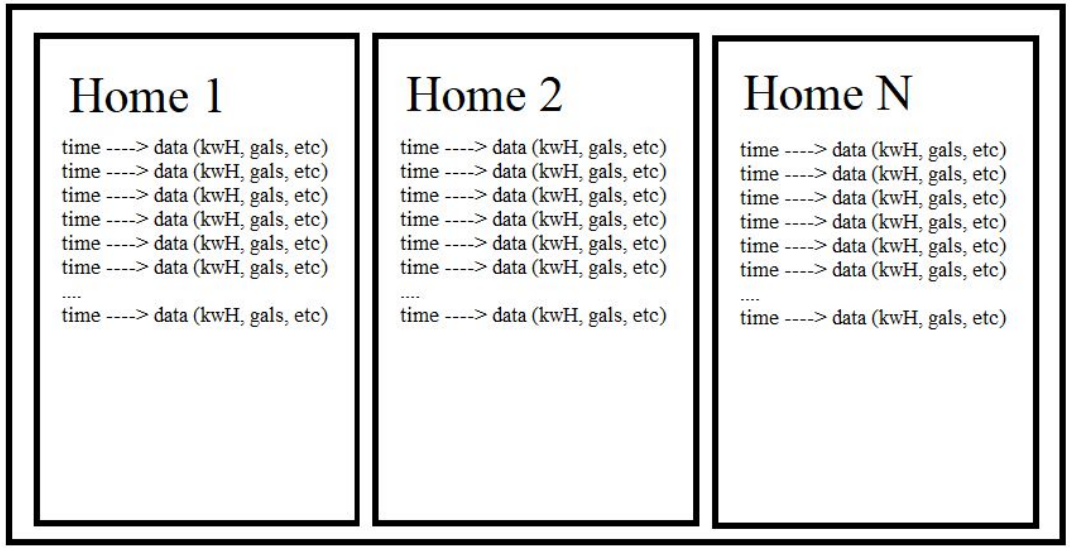
- Least Squares Algorithm
- Polynomial Regression
- Gradients
- Inverse Matrices



```
import numpy as np
import scipy

def polyreg(x,y,order):

    # We have an error function S(a0,a1,...,a_order) where "order" is the
    # degree of the polynomial that best describes the data being passed
    # into the polyreg function.

    # A polynomial is in the form y = a0 + a1*x + ... + a_order*x^order

    # Therefore, S(a) = sum( (y - (a0 + a1*x + ... + a_order*x^order) )^2 )

    # We can turn this into a linear system of equations by taking the
    # gradient of the error function S. We can then construct the
    # constant matrix "A" and the RHS vector "b" to solve for the solution
    # vector "a".

    # A = [ c0 c1 c2 . . . cOrder ]   Where ci = sum( x^i )
    #     [ c1 c2 . . .     .  .  ]
    #     [ c2 .  . .   .     .  ]           and
    #     [ .  .  .  .        .  ]
    #     [ .  .  .     .     .  ]   b = [b0 b1 ... bOrder]'
    #     [ .  .           .  .  ]
    #     [ cOrder . . . cOrder*2 ]   Where bi = sum( y * x^i )

    # We can rewrite A as:
    # A = [ cOrder . . . c2 c1 c0 ]
    #     [ .   .     . . . c2 c1 ]
    #     [ .      .     . . . c2 ]
    #     [ .         .        .  ]
    #     [ .      .     .     .  ]
    #     [ .   .           .  .  ]
    #     [ cOrder*2 . . . cOrder ]
```

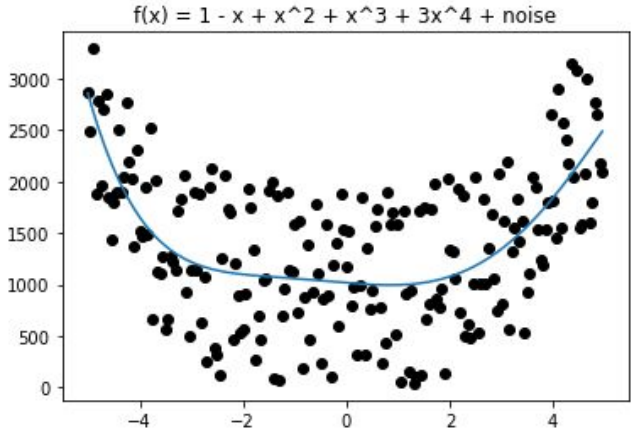
Our Successes!

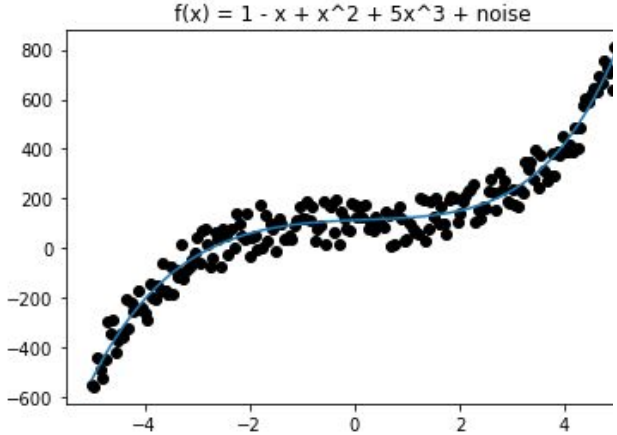
- We developed an algorithm for cross checking the datasets to find points that had the same timestamp in all three sets, sort by home, and then output relevant information to a JSON file for easy access



Our Successes!

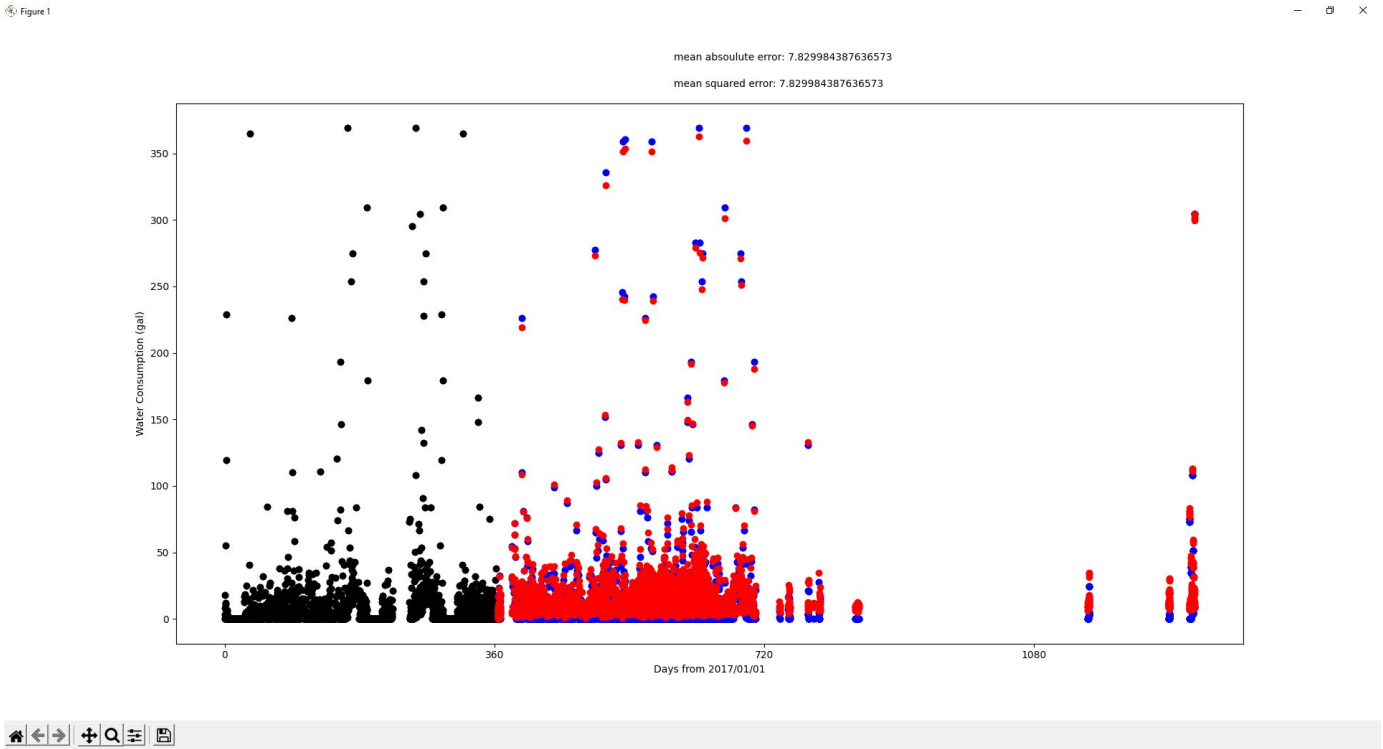
- We were able to classify and organize our data-points
- We created our own polynomial regression model for single variable functions
- We were able to visualize our findings to get a better understanding of any correlations that may exist





Success?

Black is water consumption before 2018 used for training.
Blue is actual data from after 2018 used for testing
Red is predicted data from machine learning



Findings

- We found no notable, single-variable correlation between electricity consumption and water consumption
- We believe the correlation between the two is strictly multivariable

Thank You!