The goal of this exercise is to demonstrate your machine learning skills on a realistic data set. You should spend some time writing code and documentation, but not get too carried away with spending hours tuning the solution.

Data

Two data sets are provided, both in the same format:

* common.csv -- this contains 10000 examples from a home where leaks are common so that we have about 100 examples of leaks
* rare.csv -- this contains 50000 examples from a home where leaks are rare so that we have only 3 examples of a leak.

Each row of data is an example of an event where water was flowing. The data in each column contains aggregated information about the event.

The columns in the data are as follows:

* time -- wall time for the start of the event (in UTC)
* day -- day of the week for the start of the event
* duration -- total number of seconds of water flow
* flow\_rate -- average gallons per minute
* variability -- a unitless normalized measure of the variability of the flow rate
* isleak -- ground-truth label

*Hints:*

You may assume that the rows are statistically independent from each other.

The home in common.csv has a different behavior pattern than rare.csv

Task 1

Explore the data and produce a couple plots that visualize it in an informative way. Document your approach to feature engineering: what preprocessing, if any, will you do with the data?

Task 2

Build a leak-detector for the data in common.csv. Please provide source code which does the following:

* trains on the data in common.csv as needed
* reads a second csv file in the same format, and labels that data using the trained model
* prints out metrics indicating the performance of the leak detector on the second csv input data

Task 3

Build a leak-detector for the data in rare.csv. Provide source code in the same format as in Task 2.

*More hints:*

These exercises are meant to test your ability to build a reasonable/typical first-draft end-to-end algorithm for the leak detectors. In grading your solutions, more emphasis will be placed on clear documentation of approach and reasonableness/justification of approach, and less emphasis on the absolute performance. (Although you should achieve some level of performance.) Please do not spend exorbitant amounts of time doing hyperparameter tuning or building complicated pipelines.