# Automated measurement and verification

This software trains several different regression models and chooses the best one for predicting energy use. We assessed the following regressors and use just two to reduce the amount of compute time for this particular application to a large dataset of buildings:

* (not used) K Nearest Neighbour
* (not used) Support Vector Regressor
* (not used) Gradient Boosting Regressor
* Random Forest Regressor
* Extra Trees Regressor

The software works as follows:

* Obtain training data from the input file (the only required command line argument). This data should match the format of the example file, and should contain:
  + Input data - a sequence of date-times in chronological order, with a consistent, regular interval, and with no missing date-times
  + Target data - a load value for the majority of those date-times (missing values can be blank, or ‘NA’)
  + Additional optional input data - an outside air temperature, or other weather data.
* Obtain prediction data from a prediction file. If no prediction input file is provided, the software splits the input file into two datasets: a prediction and a training set. The prediction set is held out - it is only used to evaluate the final model. No data from the prediction set is used to train models or select the best model. In this case, the default size of the prediction set is ¼, and is selected from the end of the file (forward prediction in time).
* Generate training features from the input data:
  + from the date-times, extract the minute, hour, day of week, month (if 12 months of data or more is available applicable), and federal holiday data.
  + from the weather data if present (i.e. historical data for each sample, typically 6 values evenly spread across the last 24 hours).
* Remove data with missing values.
* Normalize each input feature and the target data to a mean of 0 and a standard deviation of 1.
* Assess 5 different modeling approaches (KNR, SVR, RFR, GBR & ETR). For each:
  + Select a parameter set based on a random grid search through a defined parameter space for each regression model.
  + Perform k-fold cross-validation. Default value is k=5.
  + Fit the model k times to k-1 folds of data using the same parameter set.
  + Calculate the mean fit across the k folds for that parameter set.
  + Identify the best parameter set from Z searches in the parameter space. The number of searches (Z) varies depending on runtime per search.
* Choose the best model from each of the 5 different modeling approaches (based on the best k-fold cross validation score)
* Apply model to prediction data and:
  + If a prediction input file was provided, write a file containing the prediction data.
  + If no prediction input file was provided, assess the selected model against the held out prediction data set from the initial file input.

Dependencies

Python 2.7 with the following external packages: numpy, scipy, sci kit learn

## >>> pip install -U numpy >>> pip install -U scipy >>> pip install -U scikit-learn

## Usage

Read in a file containing measured energy data and make a (blind) prediction on the last ¼ of the data

>>>python bpe.py file.csv

Read in a file, make a prediction based on another input file, and save to predict\_output.csv

>>>python bpe.py file.csv –pi predict\_input.csv –po predict\_ output.csv

Many more options are described in the help documentation:

>>>python bpe.py –h

Examples:  
-p : plot the prediction results  
-s: save detail of each array used in the calculations, each of the final trained models, and the cross validation scores in the random grid search.  
-v: verbose output  
-k 10 : set the number of cross validation folds to ten.  
etc.