Alice Kish
January 2022

OP_complete1

Flow, Parameters and Guidance

This script takes information about a single device and returns ON and OFF times that minimise the 'cost' whilst also aligning to user constraints.

The code may be run in two separate states: debug and test. Debug state uses template inputs and the test state uses real inputs from an external device (eg Raspberry Pi) and API forecasts.

INPUTS

- Carbon Intensity 24hr forecast (source: https://carbonintensity.org.uk/)
- Energy Price 24hr forecast (source: https://octopus.energy/tariffs/)
- Device Information in following format shown in Table 1.
- Price weighting (integer between 0 and 1)

Table 1

Parameter	**Value**	***comments***
device_name	String eg. '*EV charger*'	
power	Integer eg. 4	*In multiples of 0.25MW*
on_period	Integer eg. 6	*In multiples of 30 minutes*
use_OFF	Boolean T/F eg 'true'	
s_OFF	String HH:MM eg. '14:30'	*Start of 'must be OFF' period, minutes must be 00 or 30*
e_OFF	String HH:MM eg. '14:30'	*End of 'must be OFF' period, minutes must be 00 or 30*
use_deadline	Boolean T/F eg 'true'	
deadline	String HH:MM eg. '14:30'	*Minutes must be 00 or 30*

OUTPUTS

- Time to turn device ON
- Time to turn device OFF
- This is returned in form of a csv file when run in 'test' mode, or simply printed to console if run in 'debug' mode

TO SELECT DEBUG OR TEST MODE

To select mode, navigate to the section entitled 'CHOOSE BRANCH OF CODE TEST/REAL DATA' and simply comment out the appropriate line.

```
# comment/uncomment one of the following to run code with test or
real data

#forecast_branch = 'test'
forecast_branch = 'real'

device_branch = 'test'
#device_branch = 'real'
```

Alice Kish
January 2022

Template CI/Price forecasts, Device Information and Price Weighting can all be adjusted within the following functions to suit debugging requirements:

make_ci()
make_price()
make_device_info()
make_pr_weight()

UNDERSTANDING USER CONSTRAINTS

There are two temporal constraints that the user can apply to the optimisation.

- **use_OFF, s_OFF, e_OFF**
  - This constraint allows the user to set a time period where the device must be off. To use this constraint, **use_OFF** is set to 'true' and the start and end times of the off period are entered into **s_OFF** and **e_OFF** respectively in the form 'HH:MM'.
- **use_deadline, deadline**
  - This constraint allows the user to set a deadline by which the device must have completed its on period. To activate this constraint, **use_deadline** must be set to 'true' and the **deadline** variable set to a time in the format 'HH:MM'.

OPTIMISATION

In general, the optimisation to find the 'best' time to turn the device ON/OFF works as follows:

1. The CI and Price forecast data is gathered into a dataframe.
2. The forecast data is normalised.
3. Weighting values are calculated as below.
   a. Price weighting (p) is an input number between 0 and 1
   b. Carbon Intensity (c) weighting is equal to (1-p)
4. For each 30 minute time period, a cost variable is created using the equation below:

$$Z = p * price + c * CI$$

5. The lowest Z value is found with its associated time.
6. This is returned as the device start time, and the end time is calculated by adding the time that the device is on to this start time.
7. The device start and end times are returned.

See Figure 1 and Figure 2 for examples of template forecasts and the associated cost variable.

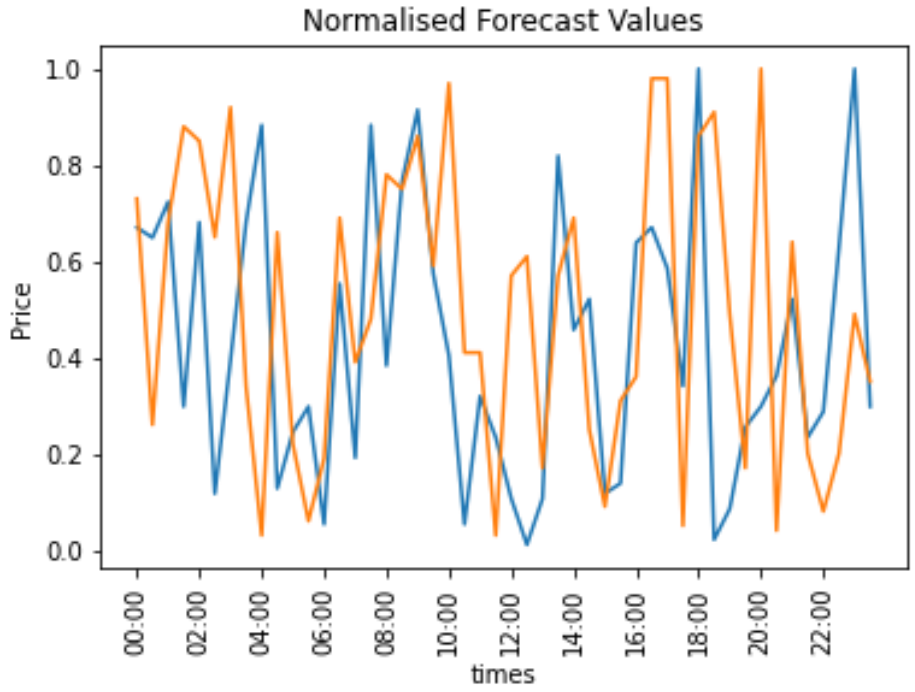

Figure 1 template CI and Price forecasts generated using random values and normalised

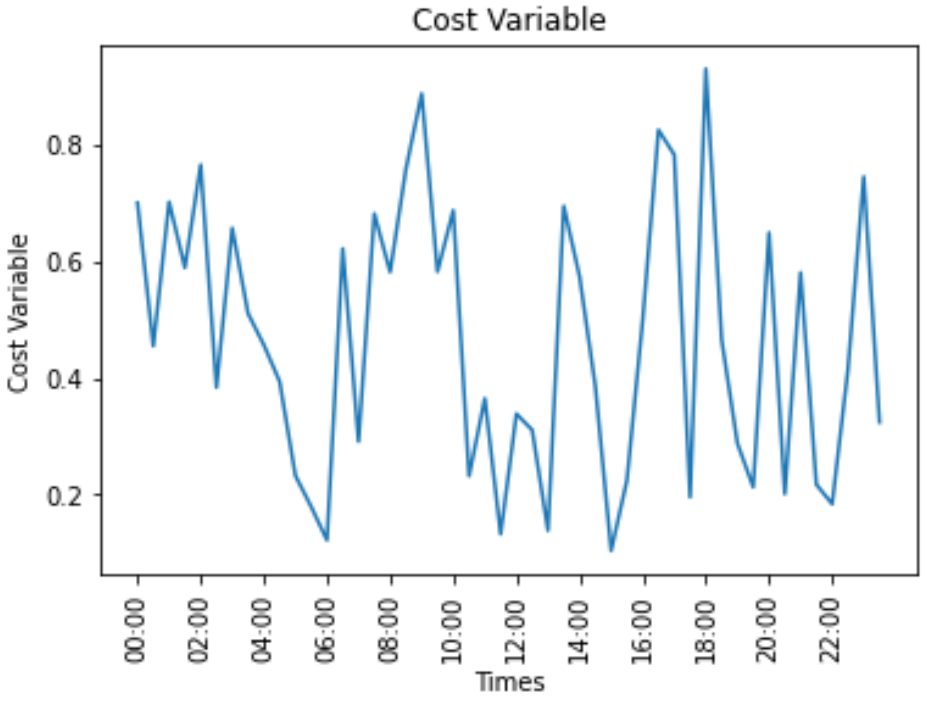

Figure 2 cost variable for the forecasts in Fig.1 with price weighting = 0.5

OPTIMISATION VARIATIONS

To take into account the different constraints, there are different optimisations available. These are shown in Table 2.

Table 2

Constraints		Optimisation	Description
use_OFF	**use_deadline**		
False	False	Op1	finds best time out of all times in forecast
True	False	Op2	Splits forecast into two cuts, before and after the 'must be OFF' period
False	True	Op3	Crops forecast at the specified deadline such that device will have turned off by the deadline
True	True	Op4	Incorporates both deadline and 'must be OFF' constraints by cropping and splitting the forecast as appropriate

Additionally, there can be complications if the forecasts received from the API are not the same length, or are truncated and don't show the full 24 hour forecast. There are checks within the code to accommodate this.