```
%{
This function calculates the timestep between each row using the
 timestamp column and
deletes the rows that are not needed from the extracted features
 table.

Arguments
- `timestampColumn`     -> the timestamp column for a single dataset
- `dataTable`           -> table containing an extracted feature (e.g.
 mean)
- `timeInterval`        -> required time interval (delta t) in
 milliseconds

Returns
- `reducedData`         -> the time domain data after reducing it
 using the
given delta t.
- `interval`            -> the number of readings between each
 interval
%}

function [reducedData, interval]=reduceData(timestampColumn,dataTable,
 timeInterval)

    % the max and min time increment steps allowed between
 consequetive
    % readings. If a step beyond the allowed limits is found an error
 is
    % raised. These are in milliseconds.
    maxStepAllowed_ms = 11;
    minStepAllowed_ms = 9;

    % find the average timestep between rows
    avg_timestep = mean(diff(timestampColumn));

    % differentiate between seconds and milliseconds and define the
 interval
    % jump based on that. The 'interval' variable will define the Nth
 row to
    % take from the extracted data
    if (avg_timestep > minStepAllowed_ms) && (avg_timestep <
 maxStepAllowed_ms)
        % timestep is in milliseconds
        interval = int16(timeInterval/avg_timestep);
    elseif (avg_timestep > minStepAllowed_ms/1000) && (avg_timestep <
 maxStepAllowed_ms/1000)
        % timestep is in seconds
        interval = int16((timeInterval/1000)/avg_timestep);
    else
        fprintf("\nError - cannot determine the timestep unit: %f\n",
 avg_timestep)
    end
```

```
    % loop through the data and only take the relevant rows (e.g.
 every fifth row)
    for ii = 1 : interval : length(dataTable)
        index = (((ii-1)/5)+1);
        incrementing_reducedData(index,:) = dataTable(index, :);
    end

    % return reducedData
    reducedData = incrementing_reducedData;
end
```

Published with MATLAB® R2020a