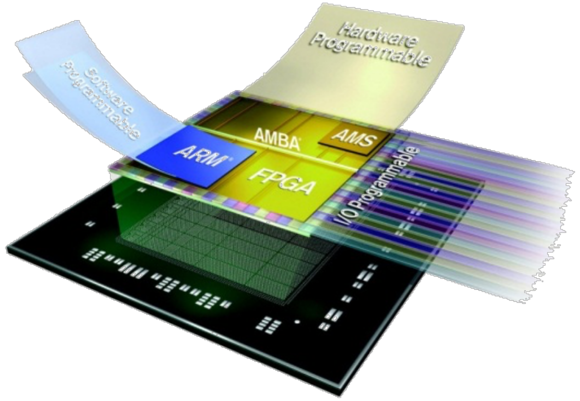
MATLAB EXPO 2016
KOREA

4월 28일 (목)

등록 하기 matlabexpo.co.kr

센서 데이터 애널리틱스를 위한 신호처리 및 머신러닝 기법

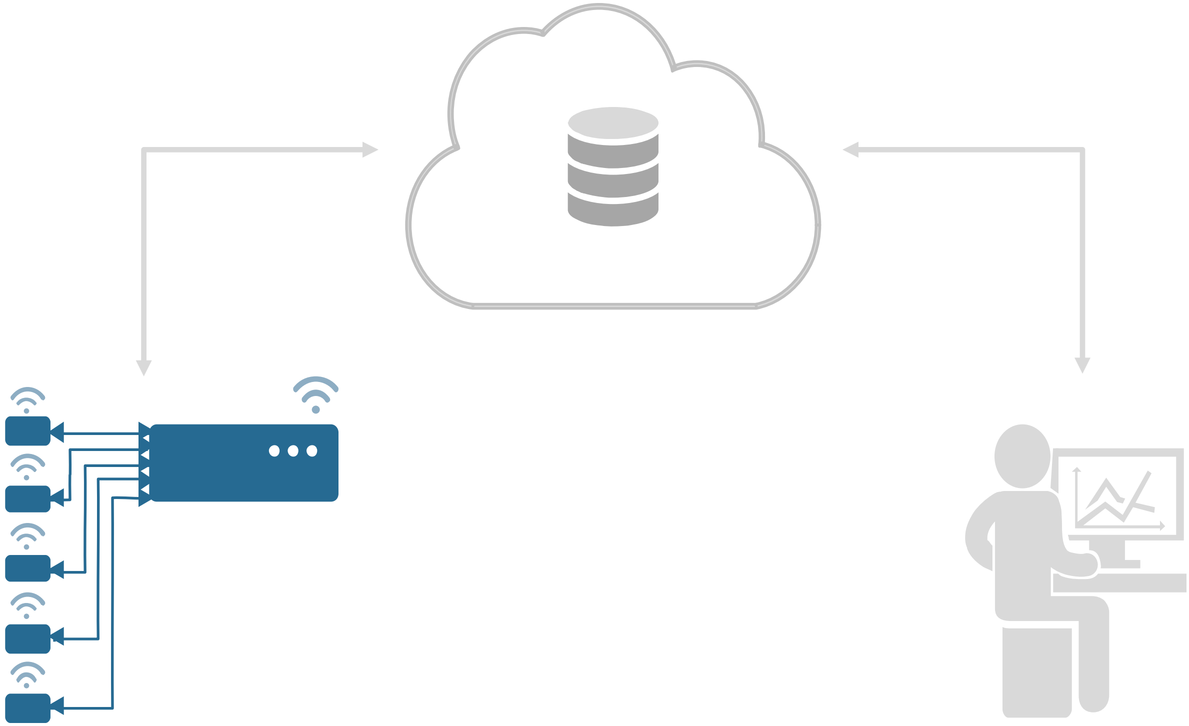


이웅재 부장

Application Engineering Group

Ryan.Lee@mathworks.com

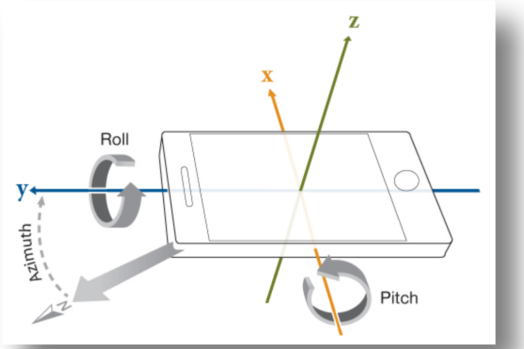
Sensor Analytics and Edge Node Development

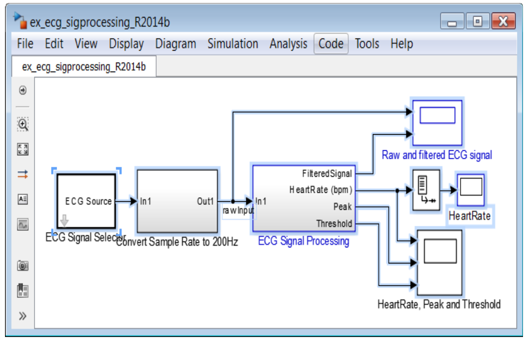


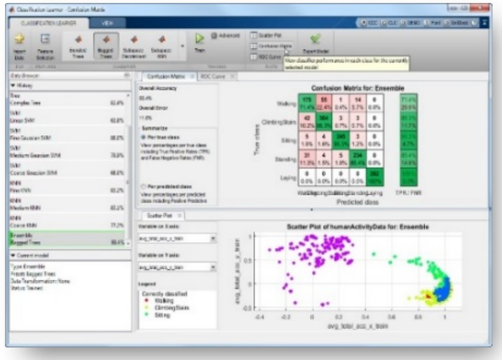
Key topics

- Signal Processing methods (e.g. digital filtering, spectral analysis)
- Machine Learning algorithms (e.g. classification)
- MATLAB environment “enablers” (e.g. language, visualization, Apps, documentation)
- Flow from predictive algorithms to embedded implementation (e.g. DSP system simulation, automatic code generation)

Sensor Analytics Workflow







Acquire

Data Acquisition Toolbox
Instrument Control Toolbox
Hardware Support Packages (XX)
Database Toolbox
ThingSpeak

Analysis

Simulink
Signal Processing Toolbox
DSP System Toolbox

Analytics

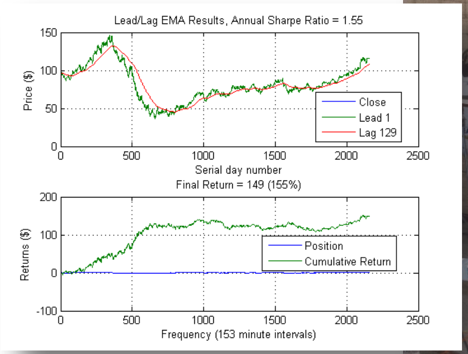
MATLAB
Stats & Machine Learning TB
Neural Networks Toolbox

Deployment

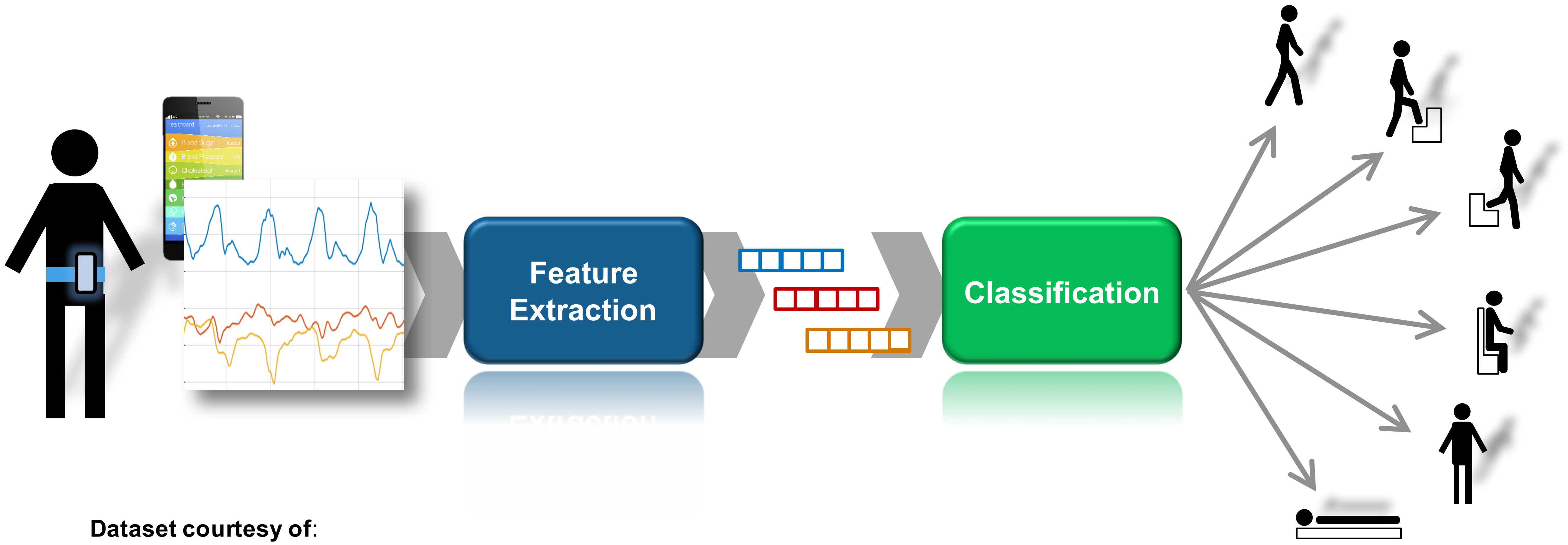
MATLAB Coder
Simulink Coder
Embedded Coder
MATLAB Compiler
MATLAB Production Server

Signal analysis for classification
Application examples

- Mobile sensing
- Structural health monitoring (SHM)
- Fault and event detection
- Automated trading
- Radar post-processing
- Advanced surveillance
- …

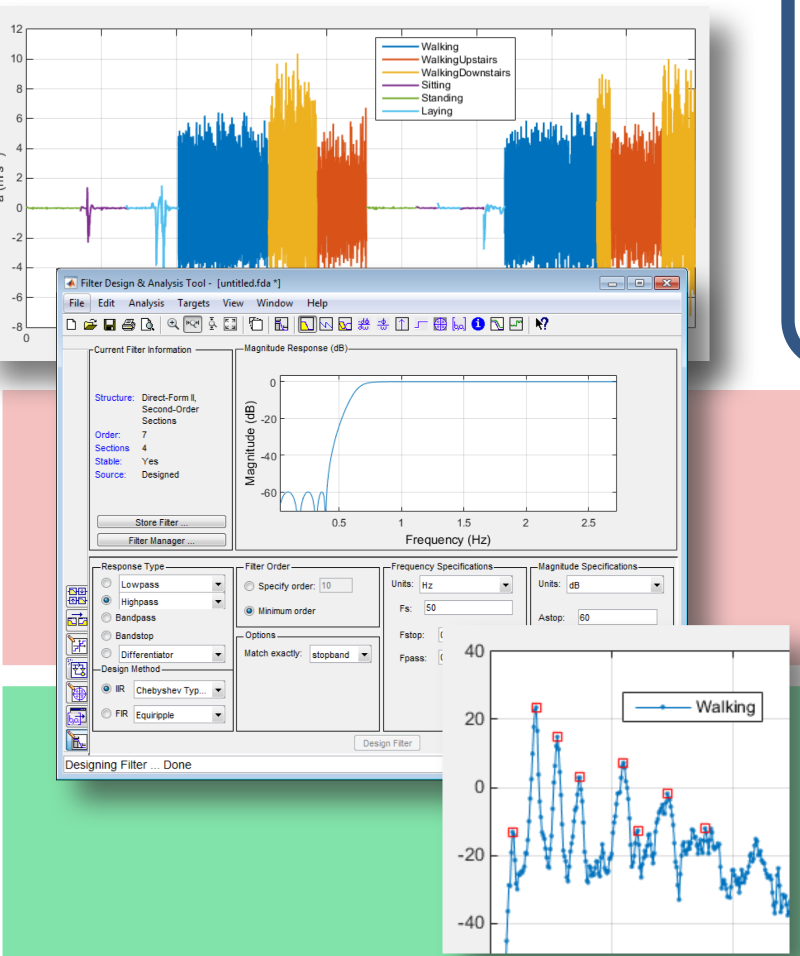


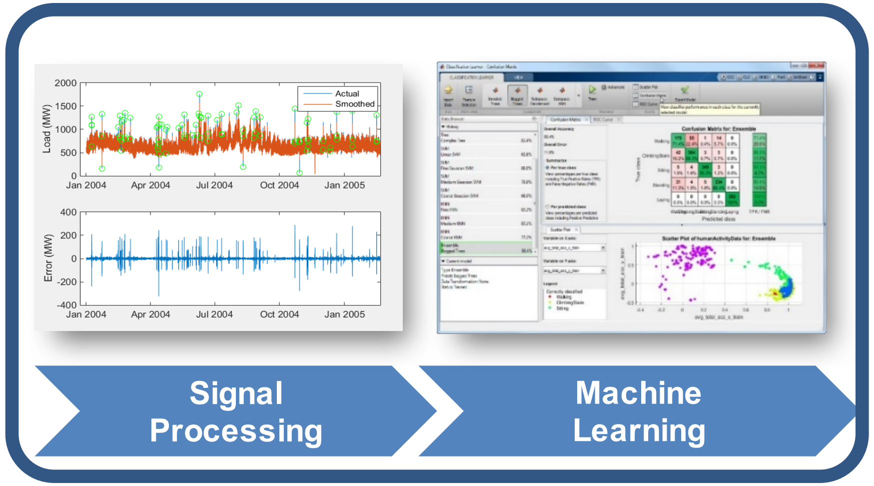
Example: Human Activity Analysis and Classification

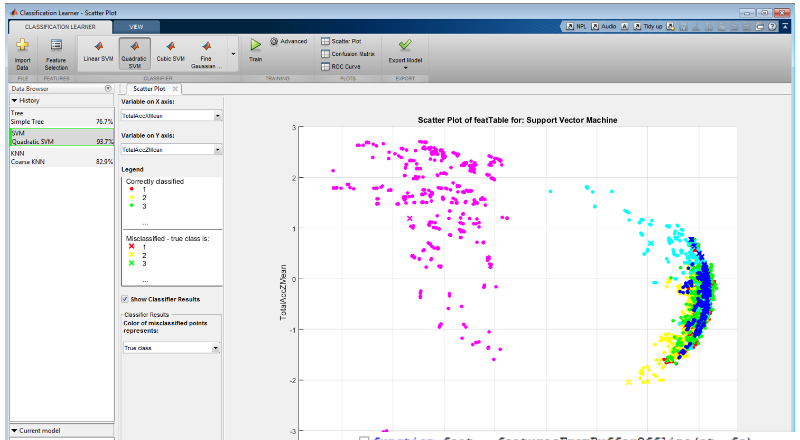


Dataset courtesy of:
Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra and Jorge L. Reyes-Ortiz.
Human Activity Recognition on Smartphones using a Multiclass Hardware-Friendly Support Vector Machine.
International Workshop of Ambient Assisted Living (IWAAL 2012). Vitoria-Gasteiz, Spain. Dec 2012
http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones

Sensor Data Analytics Workflow – the bigger picture





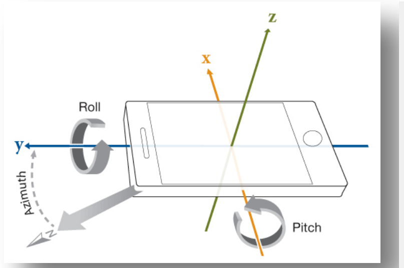


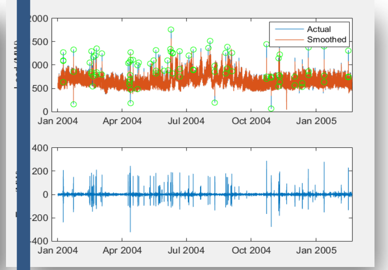

```
function feat = featuresFromBufferOffline(at, fs)
% featuresFromBuffer Extract vector of features from ra
%
% Copyright 2014-2015 The MathWorks, Inc.

% Initialize feature vector
feat = zeros(1,66);

% Average value in signal buffer for all three accelera
feat(1:3) = mean(at,1);

% Initialize digital filter
fhp = hpfilter;

% Remove gravitational contributions with digital filte
ab = filter(fhp,at);

% RMS value in signal buffer for all three acceleration
feat(4:6) = rms(ab,1);

% Autocorrelation features for all three acceleration c
% height of main peak; height and position of second pe
feat(7:15) = autocorrFeatures(ab, fs);

% Spectral peak features (12 each): height and position
feat(16:51) = spectralPeaksFeatures(ab, fs);

% Spectral power features (5 each): total power in 5 ad
% and pre-defined frequency bands
feat(52:66) = spectralPowerFeatures(ab, fs);


% --- Helper functions
function feats = autocorrFeatures(x, fs) ...

function feats = spectralPeaksFeatures(x, fs) ...

function feats = spectralPowerFeatures(x, fs) ...
```

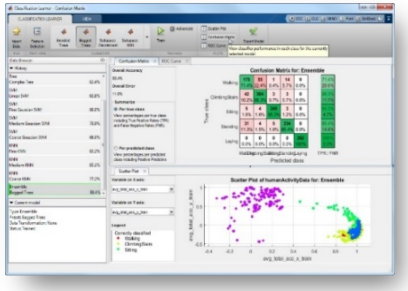
- Domain knowledge
- Open-ended problem
- Long discovery cycles

- Built-in algorithms
- Concise code (54 lines for 66 features!)
- Apps and visualisation accelerate insight

Sensor Data Analytics Workflow – the bigger picture





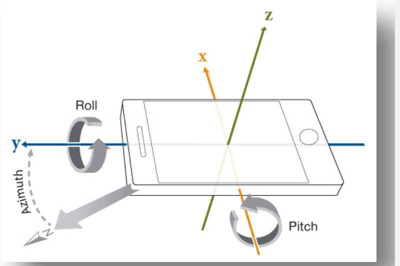


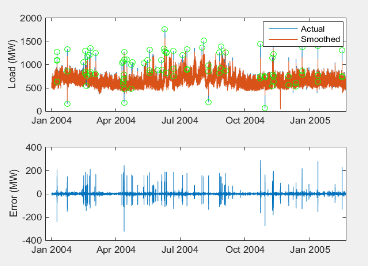
Connect and Acquire → Signal Processing → Machine Learning

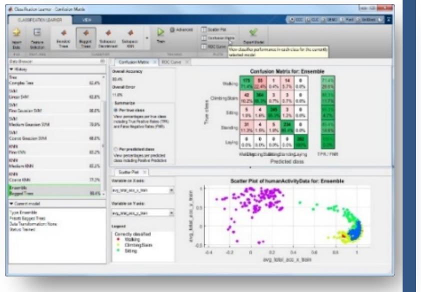
- Different tools and environments
- Disconnect between hardware and analysis
- Inefficiencies in data sharing

- MATLAB Connects to DAQ interfaces and sensors directly. E.g.
  - Android Sensor Support
  - iPhone and iPad Sensor Support

Sensor Data Analytics Workflow – the bigger picture





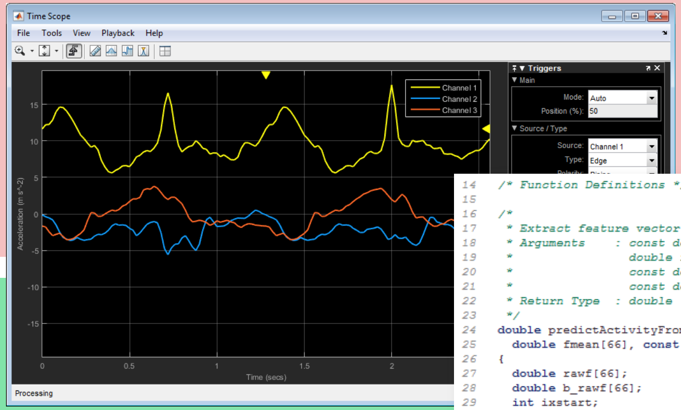


Connect and Acquire

Signal Processing

Machine Learning

Embedded Implementation




```
/* Function Definitions */

/*
 * Extract feature vector
 * Arguments    : const double at[384]
 *                double fs
 *                const double fmean[66]
 *                const double fstd[66]
 * Return Type  : double
 */
double predictActivityFromSignalBuffer(const double at[384], double fs,
  double fmean[66], const double fstd[66])
{
  double rawf[66];
  double b_rawf[66];
  int ixstart;
  double scores[6];
  double mtmp;
  int itmp;
  int ix;
  boolean_T exitg1;
  featuresFromBuffer(at, fs, rawf);

  /*  Classify with neural network */
  for (ixstart = 0; ixstart < 66; ixstart++) {
    b_rawf[ixstart] = (rawf[ixstart] - fmean[ixstart]) / fstd[ixstart];
```

- Signal analysis vs. on-line DSP
- From Machine Learning theory to pre-trained, low-footprint classifiers
- MATLAB vs. C/C++

- Streaming algorithms, data sources and visualization for System modelling and simulation
- Automatic code generation

Leverage Built-in Algorithms, Apps, and Technologies

- **Signal Processing Toolbox™**
  Built-in algorithms and Apps to process and analyse signals

$$|H(j\omega)|^2 = \frac{\varepsilon^2 C_N^2(\omega_s/\omega)}{1 + \varepsilon^2 C_N^2(\omega_s/\omega)}$$

$$\omega = 2\tan^{-1}\left(\frac{\Omega \tan\left(\pi \frac{f_p}{f_s}\right)}{2\pi f_p}\right)$$

$$C_N(\omega_s/\omega) = \begin{cases} \cos[N\cos^{-1}(\omega_s/\omega)], & |\omega| \ge \omega_s \\ \cosh[N\cosh^{-1}(\omega_s/\omega)], & |\omega| \le \omega_s \end{cases}$$

$$P_{xx}(f) = \frac{1}{LF_s}\left|\sum_{n=0}^{L-1} x_L(n) e^{-j2\pi fn/F_s}\right|^2$$

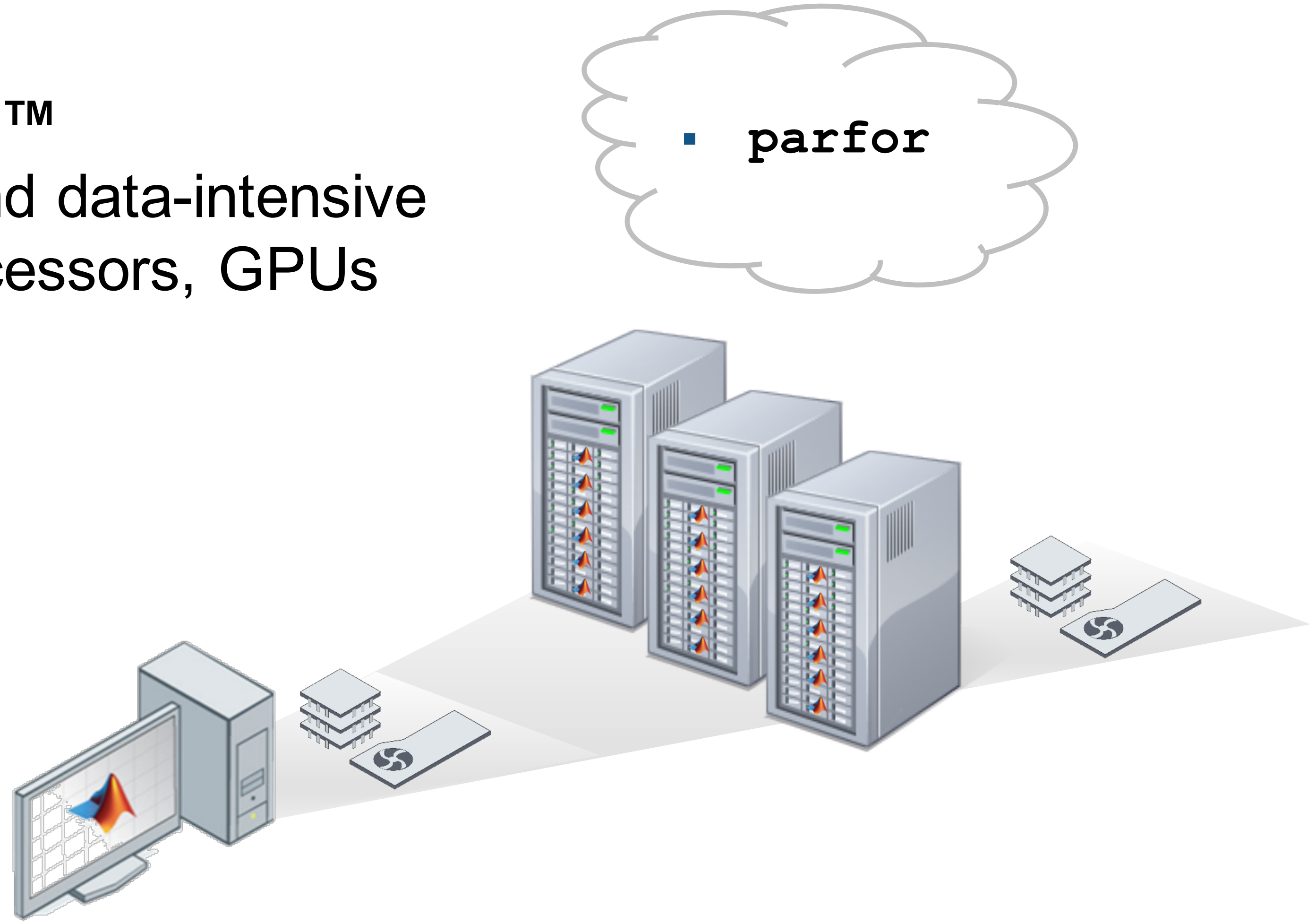
$$f_k = \frac{kF_s}{N} \quad k = 0,1,\ldots,N-1$$

$$c_{xy}(m) = \begin{cases} \sum_{n=0}^{N-|m|-1}\left(x(n+m) - \frac{1}{N}\sum_{i=0}^{N-1} x_i\right)\left(y_n^* - \frac{1}{N}\sum_{i=0}^{N-1} y_i^*\right) & m \ge 0 \\ c_{yx}^*(-m) & m < 0 \end{cases}$$

- `cheby2`
- `filter`
- `rms`
- `pwelch`
- `periodogram`
- `xcov`
- `findpeaks`
- …

Leverage Built-in Algorithms, Apps, and Technologies

- Signal Processing Toolbox™
- **Parallel Computing Toolbox™**
  Accelerate computationally and data-intensive problems using multicore processors, GPUs and computer clusters



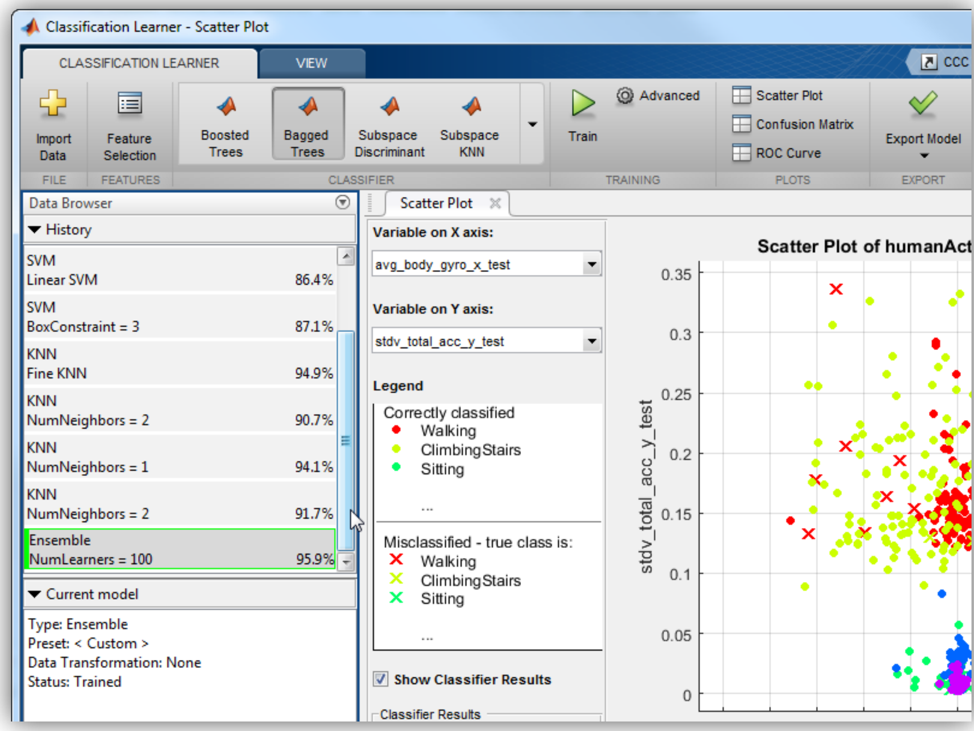
Leverage Built-in Algorithms, Apps, and Technologies

- Signal Processing Toolbox™

- Parallel Computing Toolbox™

- **Statistics and Machine Learning Toolbox™**
  Functions and apps to describe, analyze, and model data.
  Regression, clustering and classification algorithms to draw inferences from data and build predictive models

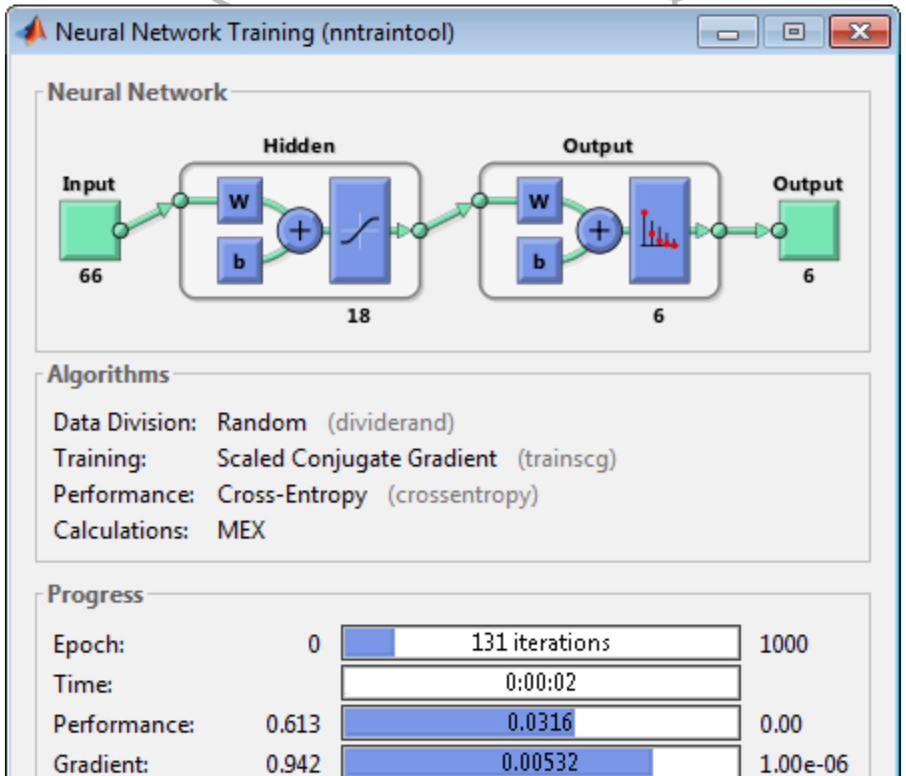
```
>> classificationLearner
```



Leverage Built-in Algorithms, Apps, and Technologies

- Signal Processing Toolbox™
- Parallel Computing Toolbox™
- Statistics and Machine Learning Toolbox™
- **Neural Network Toolbox™**
  Functions and apps to design, train, visualize, and simulate neural networks

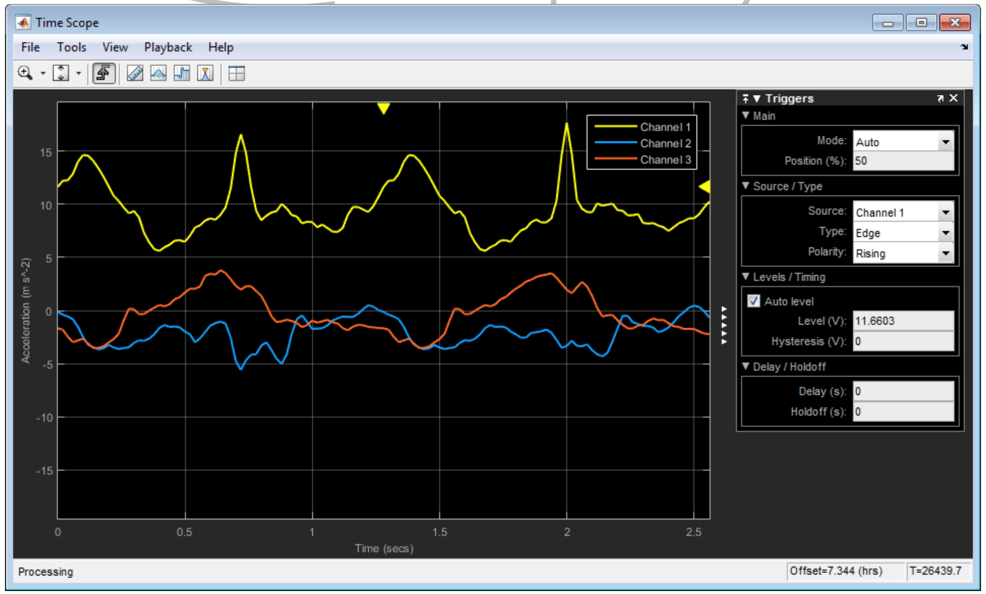
- `>> nprtool`
- `patternnet`



Leverage Built-in Algorithms, Apps, and Technologies

- Signal Processing Toolbox™
- Parallel Computing Toolbox™
- Statistics Toolbox™
- Neural Network Toolbox™
- **DSP System Toolbox™**
  Streaming algorithms, data sources and visualization for system modelling and simulation

- `BiquadFilter`
- `MatFileReader`
- `Autocorrelator`
- `SpectrumEstimator`
- `TimeScope`



Leverage Built-in Algorithms, Apps, and Technologies

- Signal Processing Toolbox™
- Parallel Computing Toolbox™
- Statistics Toolbox™
- Neural Network Toolbox™
- DSP System Toolbox™
- **MATLAB Coder™**
  Generate embeddable source C/C++ from MATLAB code
  (Learn more: [MATLAB to C Made Easy](#) webinar)

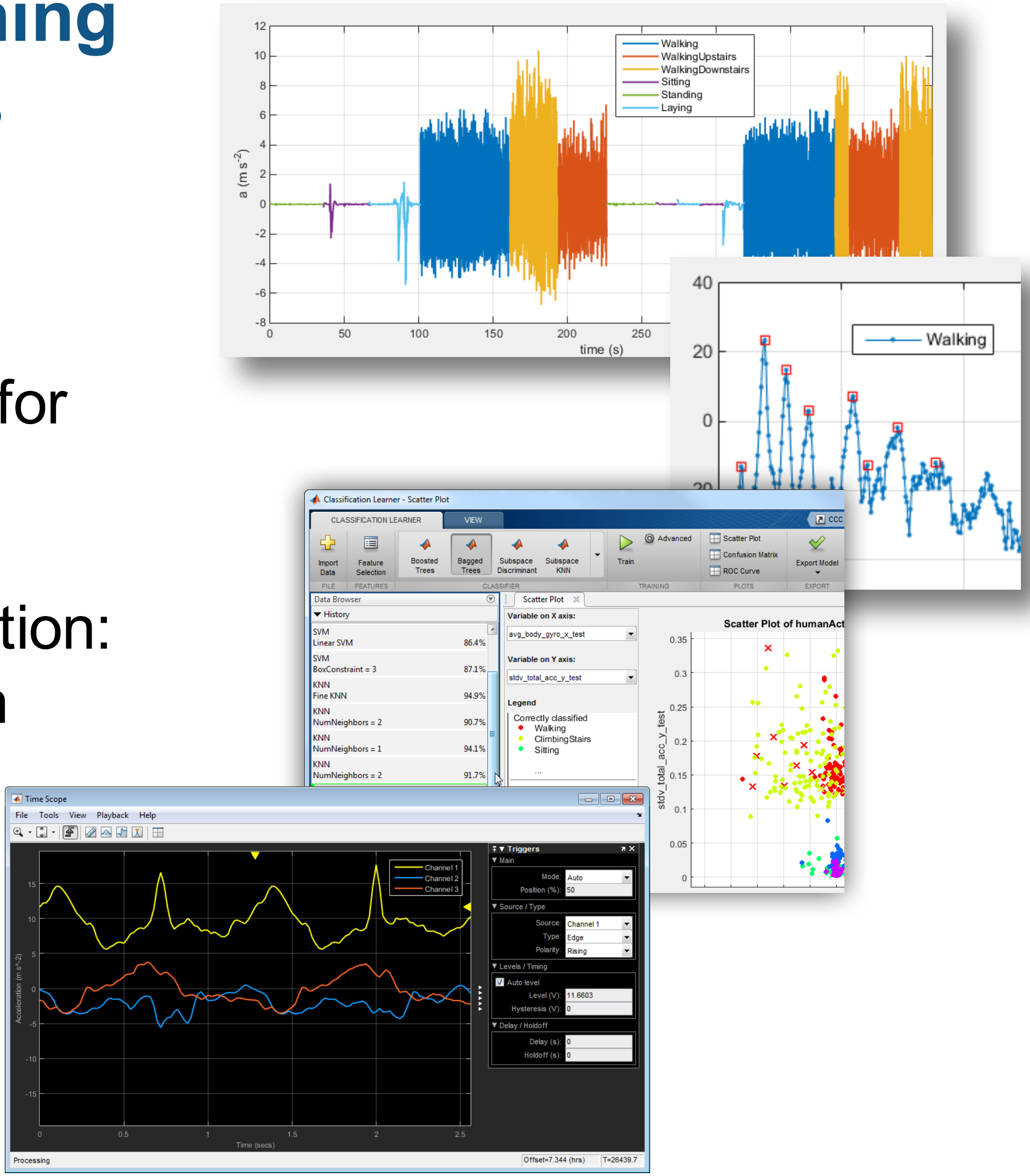
```
>> codegen
```

```
function predictedActid = predictActivityFromSignalBuffer(at, fs, fmean, fs

% Extract feature vector
rawf = featuresFromBuffer(at, fs);
f = (rawf-fmean).

% Classify with n
scores = mynn(f');
% Interpret resul
% the activity
[~, predictedActi

end
```

```
/* Function Definitions */

/*
 * Extract feature vector
 * Arguments    : const double at[384]
 *                double fs
 *                const double fmean[66]
 *                const double fstd[66]
 * Return Type  : double
 */
double predictActivityFromSignalBuffer(const double at[384], double fs,
  double fmean[66], const double fstd[66])
{
  double rawf[66];
  double b_rawf[66];
  int ixstart;
  double scores[6];
  double mtmp;
  int itmp;
  int ix;
  boolean_T exitg1;
  featuresFromBuffer(at, fs, rawf);

  /*  Classify with neural network */
  for (ixstart = 0; ixstart < 66; ixstart++) {
    b_rawf[ixstart] = (rawf[ixstart] - fmean[ixstart]) / fstd[ixstart];
```

Signal Processing and Machine Learning Techniques for Sensor Data Analytics

Summary



- Extensive set of de-facto standard functions for signal processing and machine learning
- Environment accelerates insight and automation: visualisation, apps, language, documentation
- Path to embedded products, from on-line simulation to automatic code generation