
MATLAB EXPO 2018
KOREA

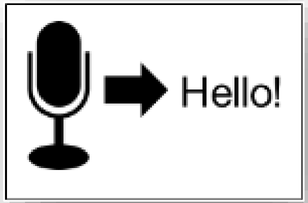
MATLAB EXPO 2018

MATLAB을 이용한 머신 러닝 (기본)

Senior Application Engineer
엄 준 상 과장

Machine Learning is Everywhere

Solution is too complex for hand written rules or equations



Speech Recognition

Object Recognition

Engine Health Monitoring

learn complex non-linear relationships

Solution needs to adapt with changing data

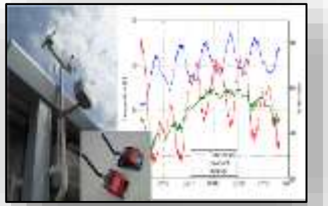
Weather Forecasting

Energy Load Forecasting

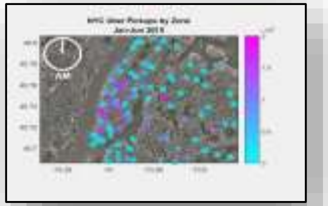
Stock Market Prediction

update as more data becomes available

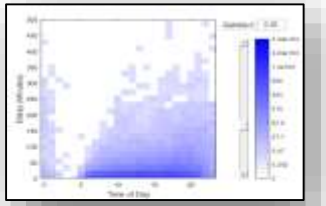
Solution needs to scale



IoT Analytics



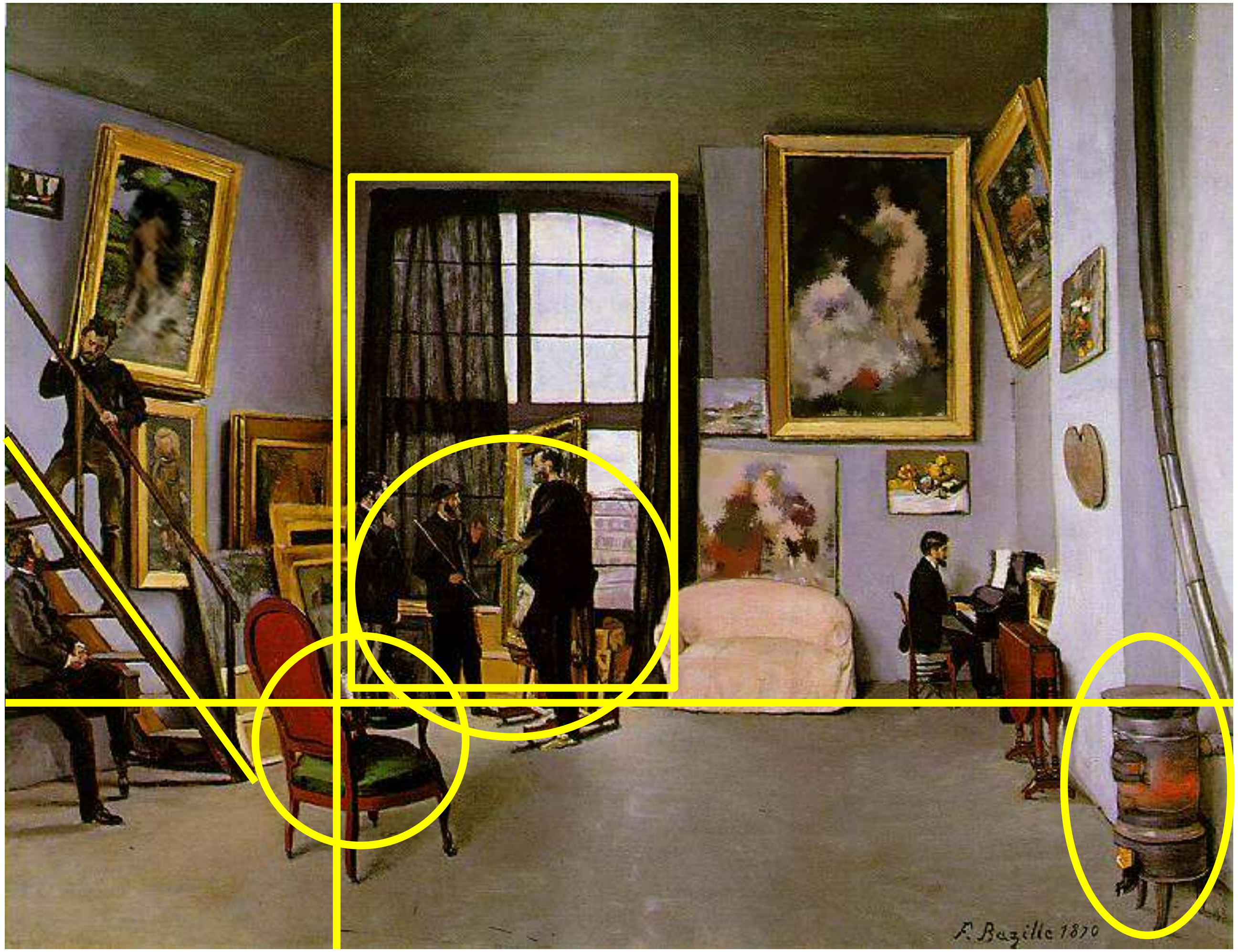
Taxi Availability



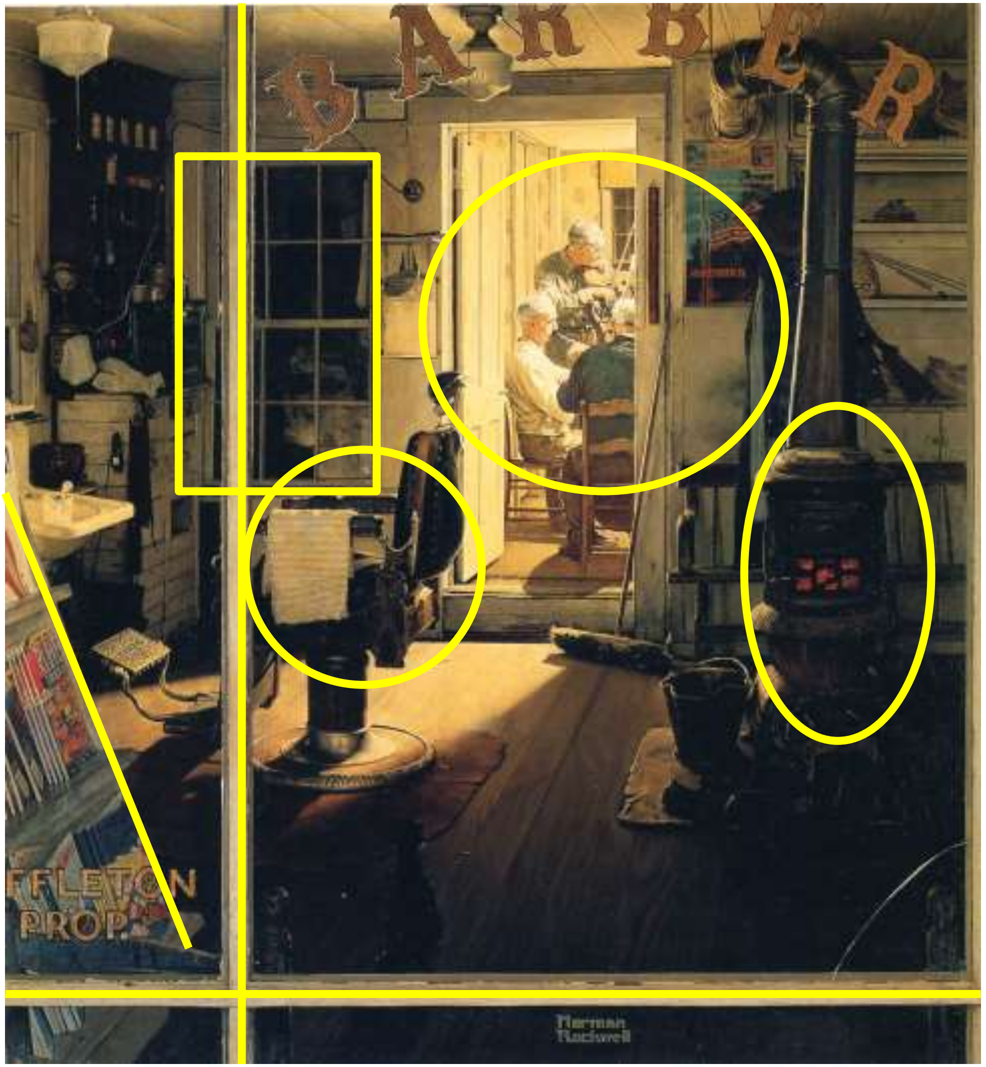
Airline Flight Delays

learn efficiently from very large data sets

MATLAB EXPO 2018



Bazille's Studio
Bazille 1870



Shuffleton's Barbershop
Rockwell 1950

Artistic Style Classification

Image Feature Extraction

Visual Features

Machine Learning Classification

Style Classifier (SVM)

Genre Classifier (SVM)

Artist Classifier (SVM)

Style: Regionalism

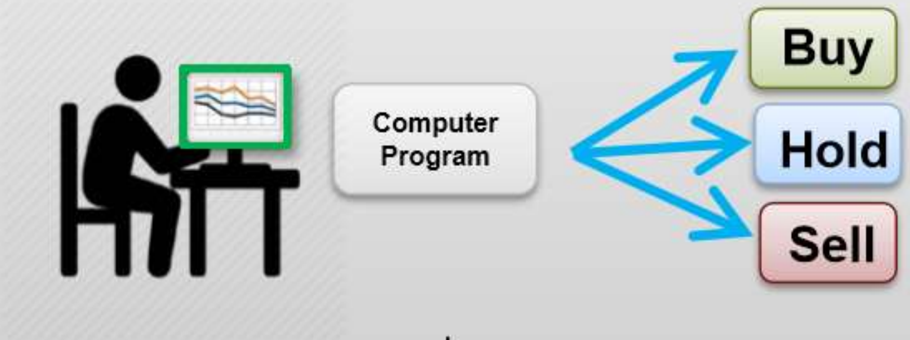
Genre: Interior

Artist: Rockwell

Machine Learning

Machine learning uses **data** and produces a **program** to perform a **task**

Standard Approach



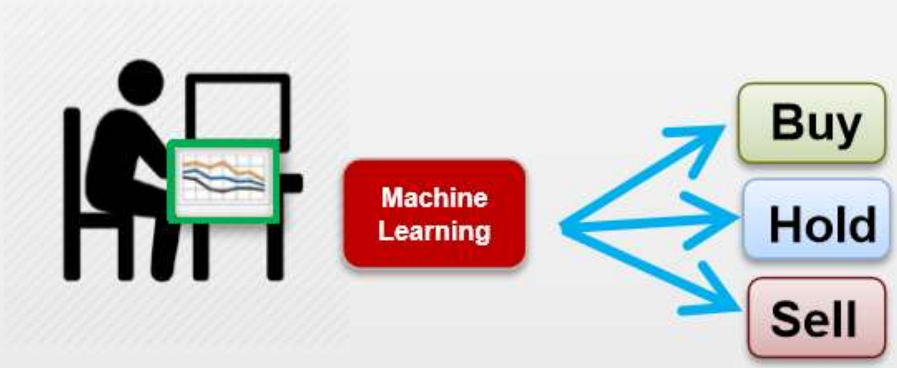
Hand Written Program

If RSI > 70
then “SELL”
If MACD > SIG and RSI <= 70
then “HOLD”
…

Formula or Equation

$$Y_{\text{Trade}} = \beta_1 X_{\text{RSI}} + \beta_2 X_{MACD} + \beta_3 X_{\text{TSMom}} + \dots$$

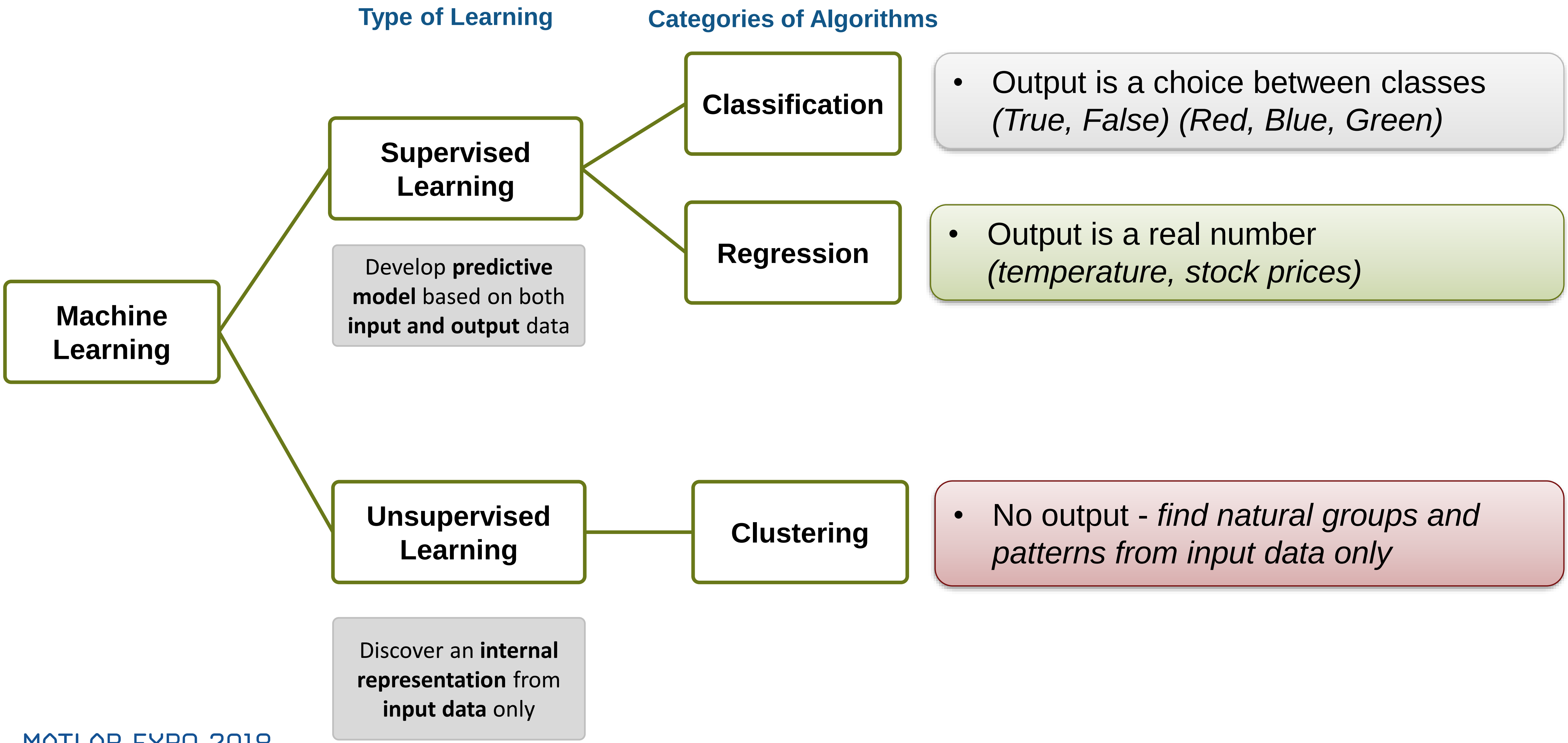
Machine Learning Approach



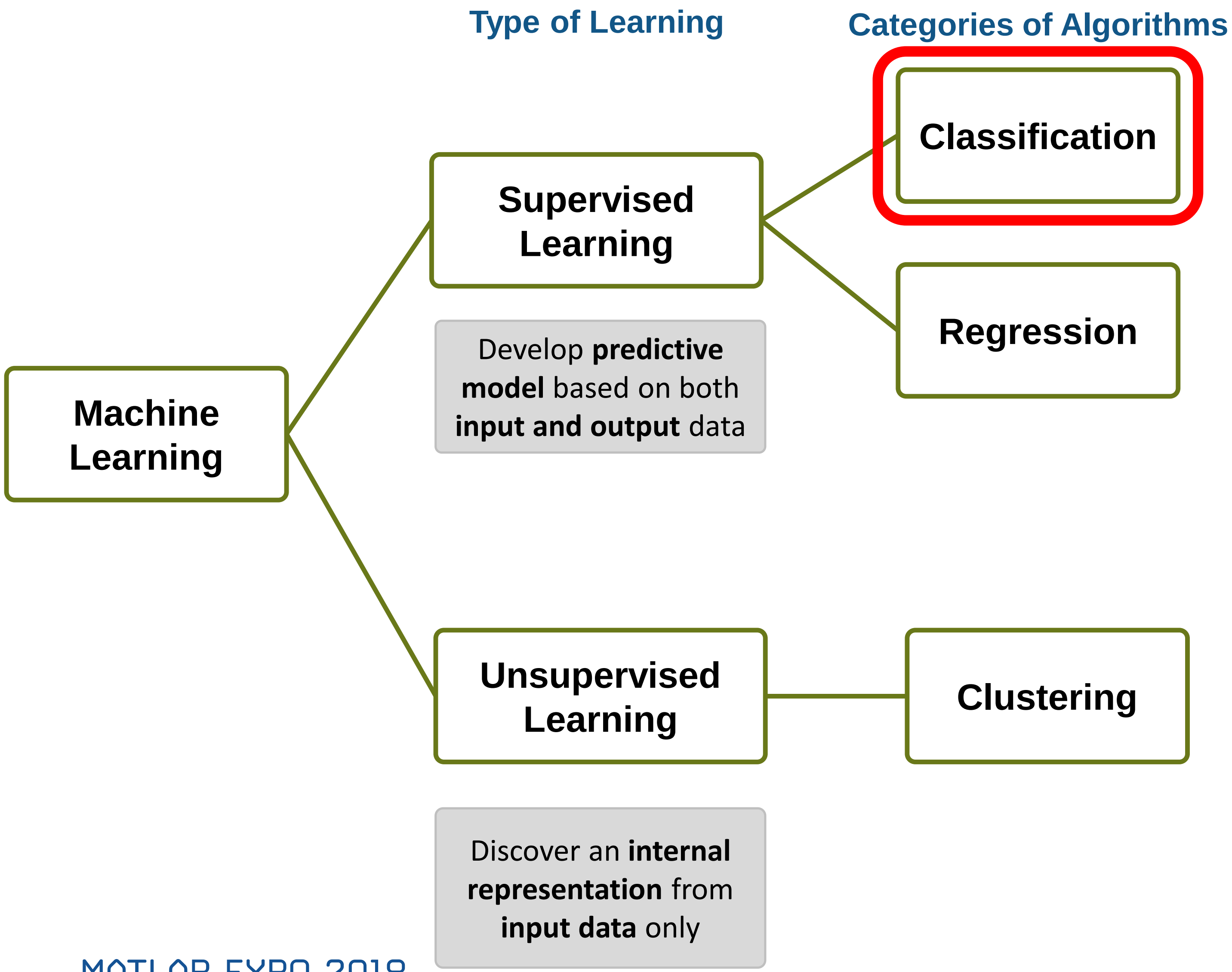
Inputs → Outputs

$$\text{Prediction} = \text{F}(\textit{factors}, \textit{trade decision})$$

Different Types of Learning

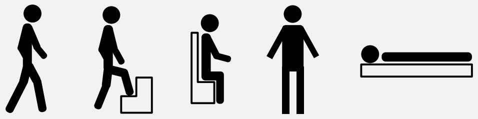


Example: Classification



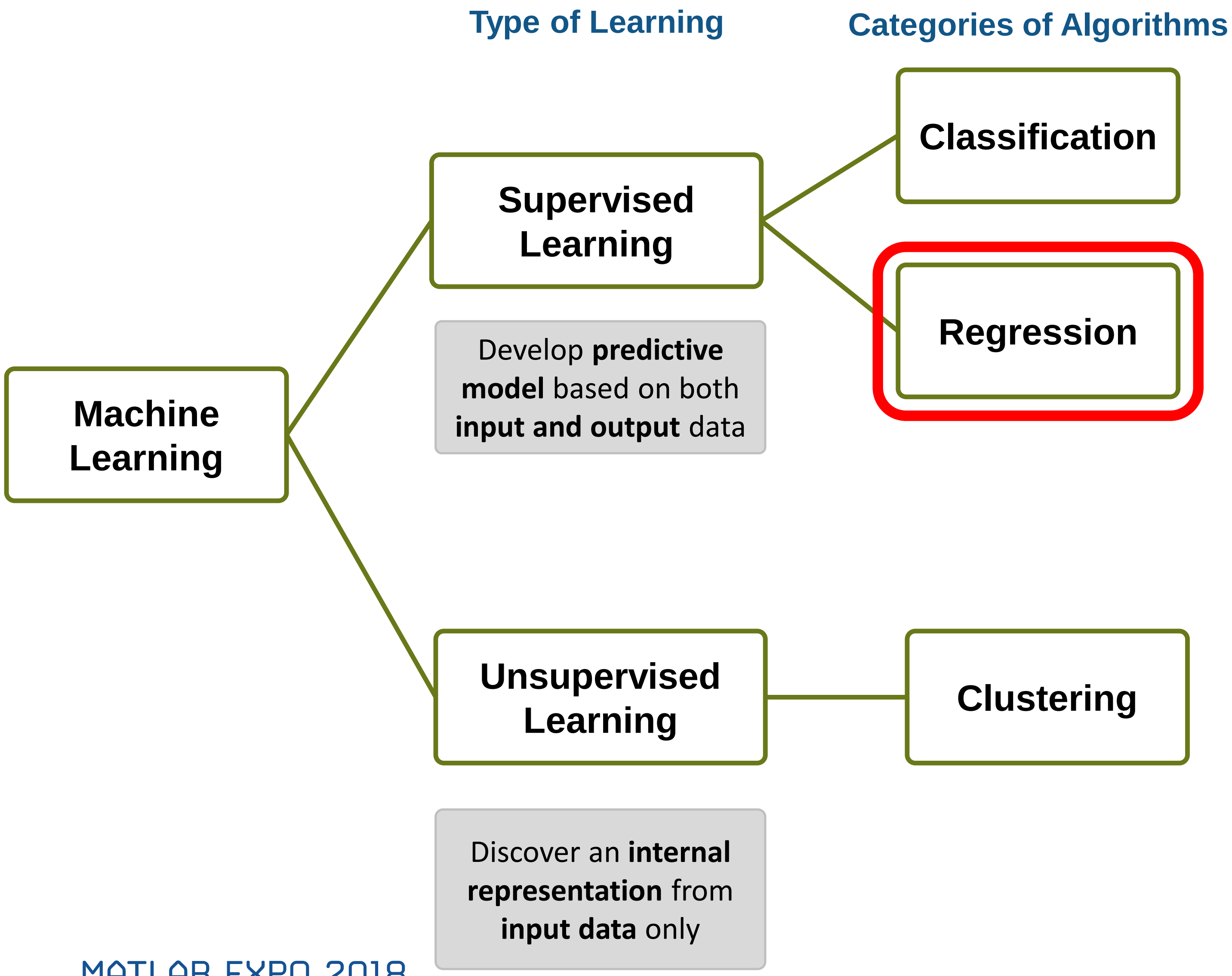
Objective:
Train a classifier to classify human activity from sensor data

Data:

Inputs	3-axial Accelerometer 3-axial Gyroscope 
Outputs	

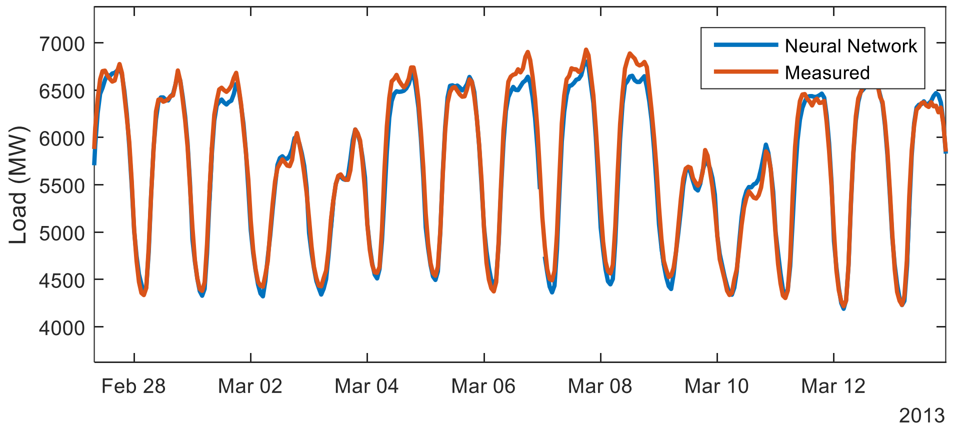
Approach:
- Import data
- Interactively train and compare classifiers
- Test results on new sensor data

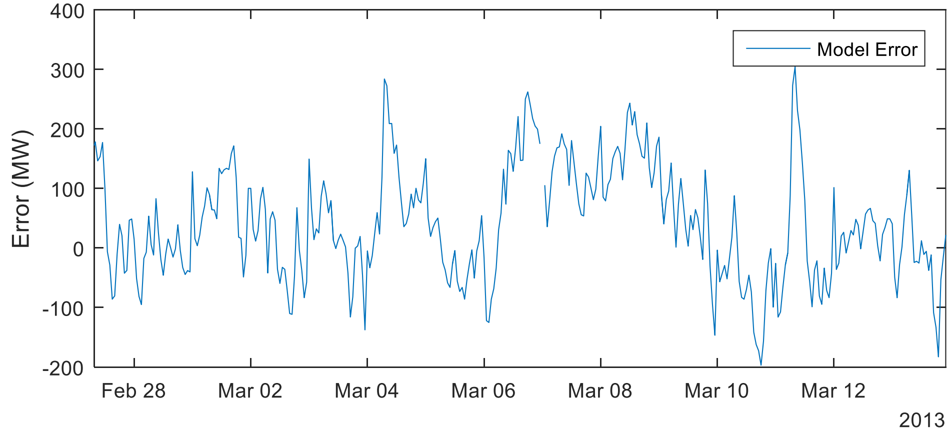
Example: Regression



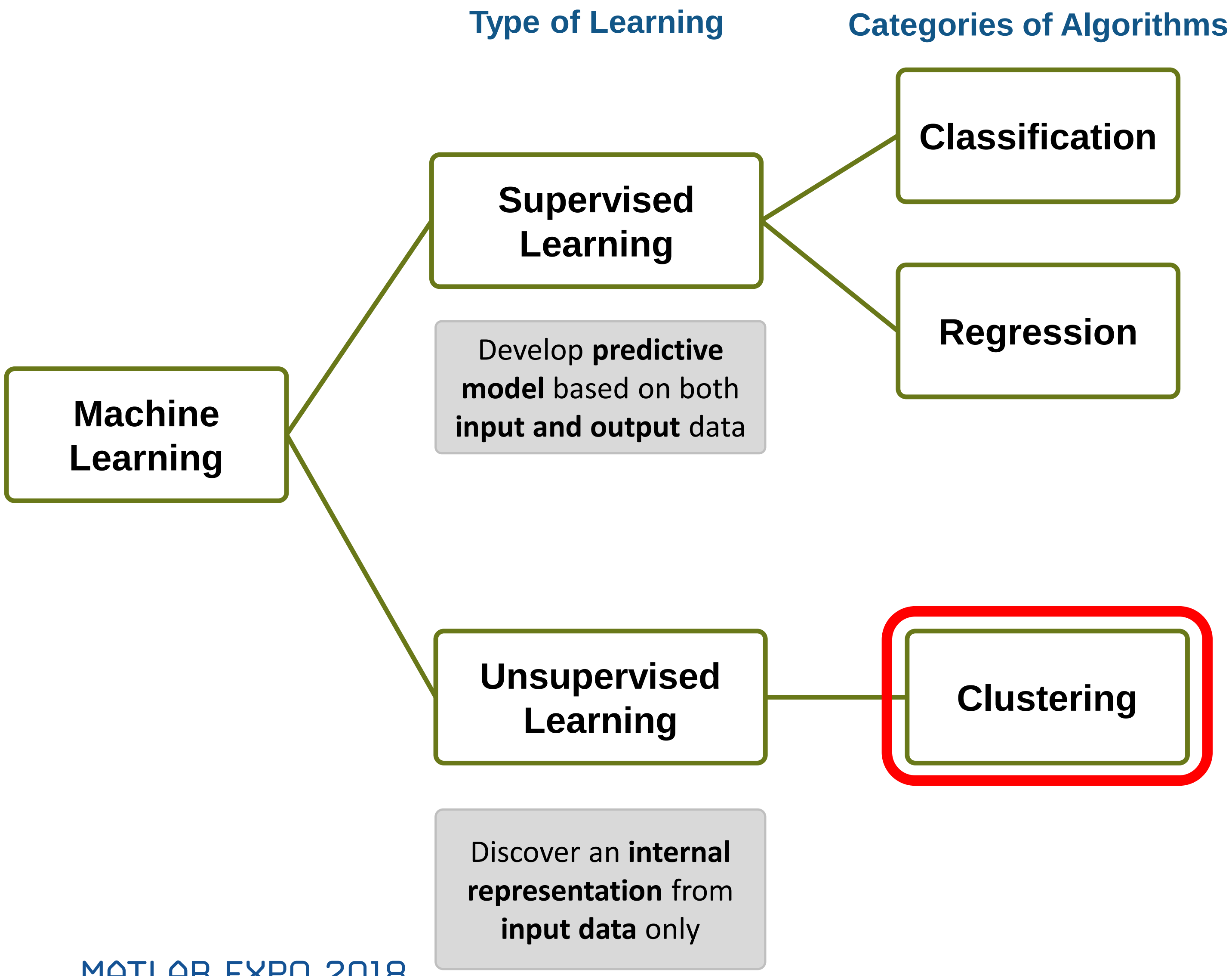
Objective:

Easy and accurate computation of day-ahead system load forecast



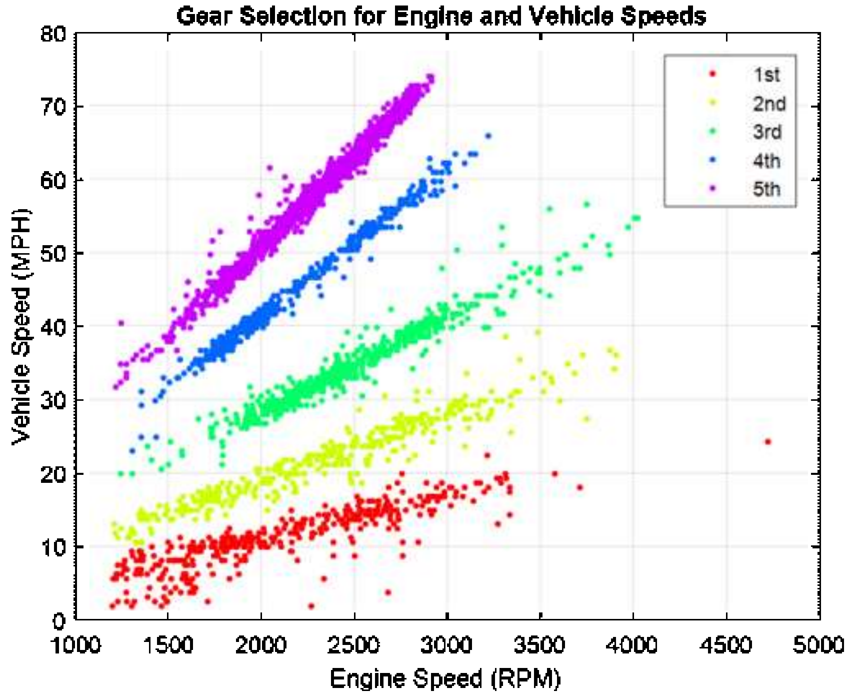


Example: Clustering



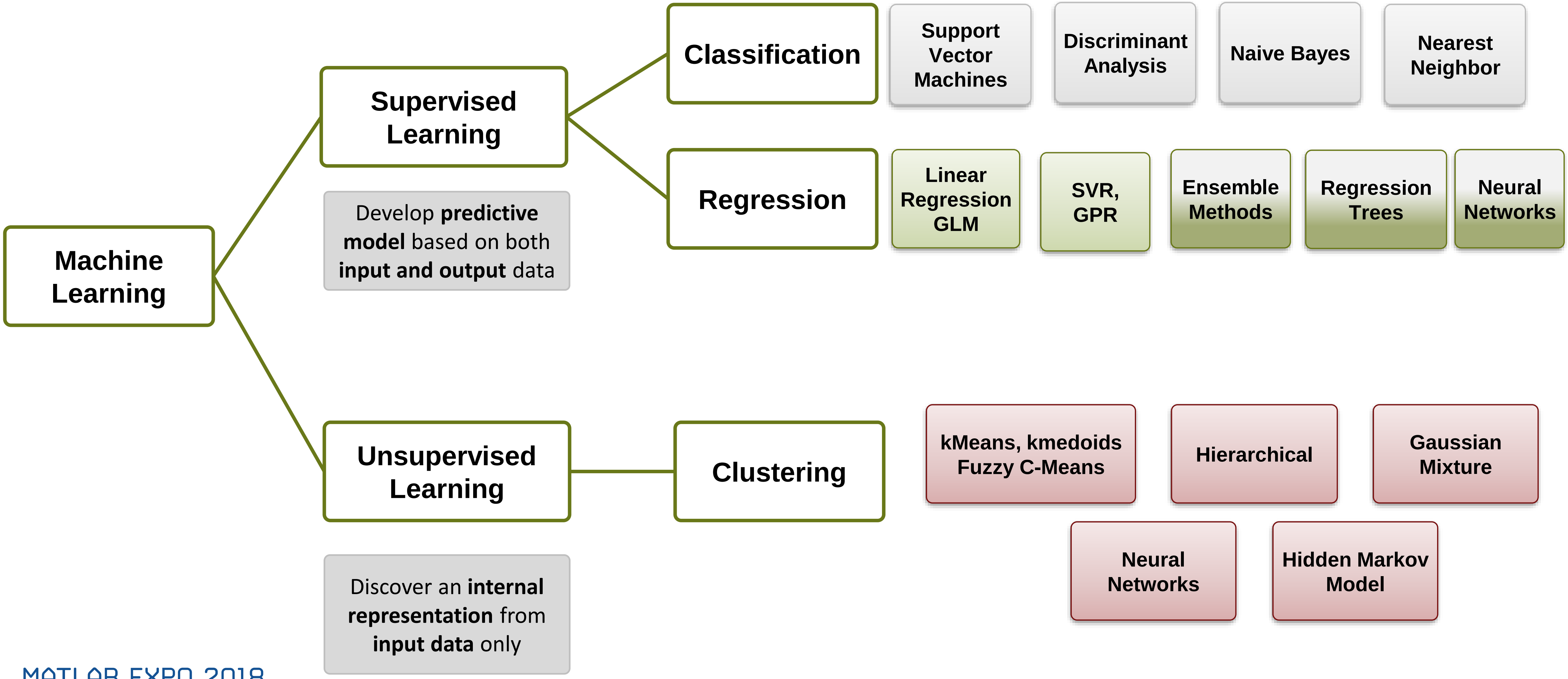
Objective:

Given data for engine speed and vehicle speed, identify clusters



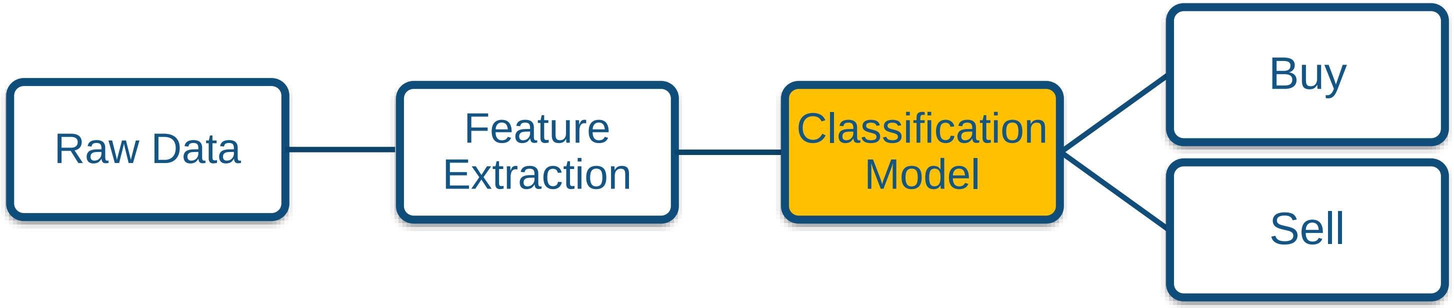
Different Types of Learning

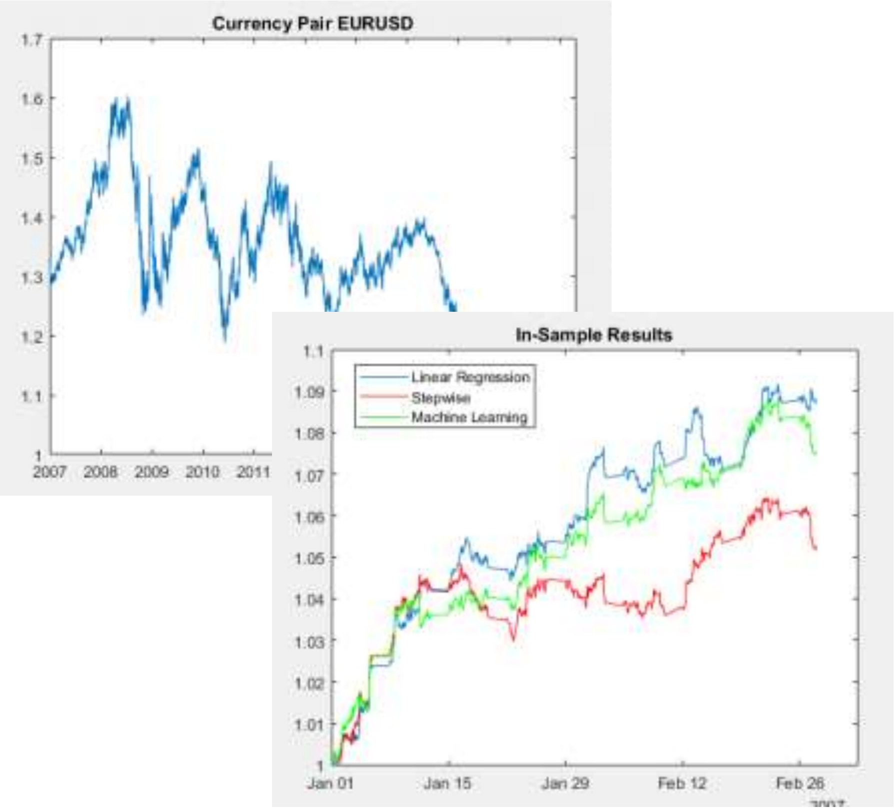
Type of Learning | Categories of Algorithms



Case Study
Machine learning techniques for algorithmic trading

- Goal
  – Developing a trading strategy





- Data
  – Multiple factors
  – Based on fundamentals or price data
  – Tested on historical data

The Challenge

- Persona: FX Trader
- Question: Can we predict the future price/return of a currency pair
  - E.g. 60 minutes into the future
- Using: Historical intra-day data
  - Recent returns
  - Technical Indicators
- Creating: A predictive model
  - Regression / machine-learning
  - Backtest over a suitable period of time

Data

- Currency Pair: EURUSD
- Data: Ten years of one-minute bar prices
  - bid/ask/mid
- Stored: In **`timetable`** objects

10×3 timetable

Time	Mid	Bid	Ask
01-Jan-2007 00:00:00	1.31916	1.31908	1.31924
01-Jan-2007 00:01:00	1.31929	1.31921	1.31937
01-Jan-2007 00:02:00	1.31954	1.31946	1.31962
01-Jan-2007 00:03:00	1.31963	1.31958	1.31968
01-Jan-2007 00:04:00	1.31952	1.31945	1.31959
01-Jan-2007 00:05:00	1.3195	1.31942	1.31958
01-Jan-2007 00:06:00	1.31945	1.3194	1.3195
01-Jan-2007 00:07:00	1.31965	1.31962	1.31968
01-Jan-2007 00:08:00	1.31958	1.31953	1.31963
01-Jan-2007 00:09:00	1.319525	1.31945	1.3196

Factor Creation - Predictors

- A mixture of factors from the Financial Toolbox and hand written
- Toolbox
  - `rsindex` (5, 10, 15, 20, 25, 30 & 60 minute)
  - `macd`
- Derived
  - N-Minute return (5, 10, 15, 20, 25, 30 & 60 minute)

The Trading Model

- Train a model based around a number of factors
  – Technical Indicators & Short Term Returns
  – Attempt to predict positive or negative future returns using current information
  – Trade on this prediction

- Model Selection
  – Linear regression and stepwise
  – Classification Tree

- Backtesting
  – Test over ten years, taking into account bid/offer spread as trading cost
  – Varying our length of in-sample and out-sample

Step1 : Data Regression

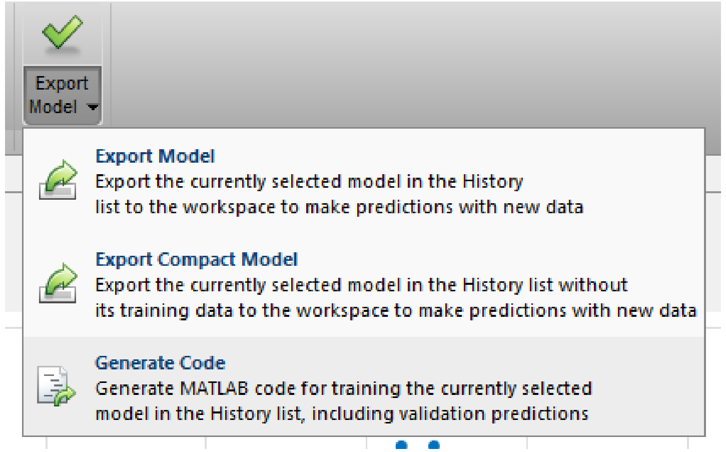
- Continuous Data to Discrete Data

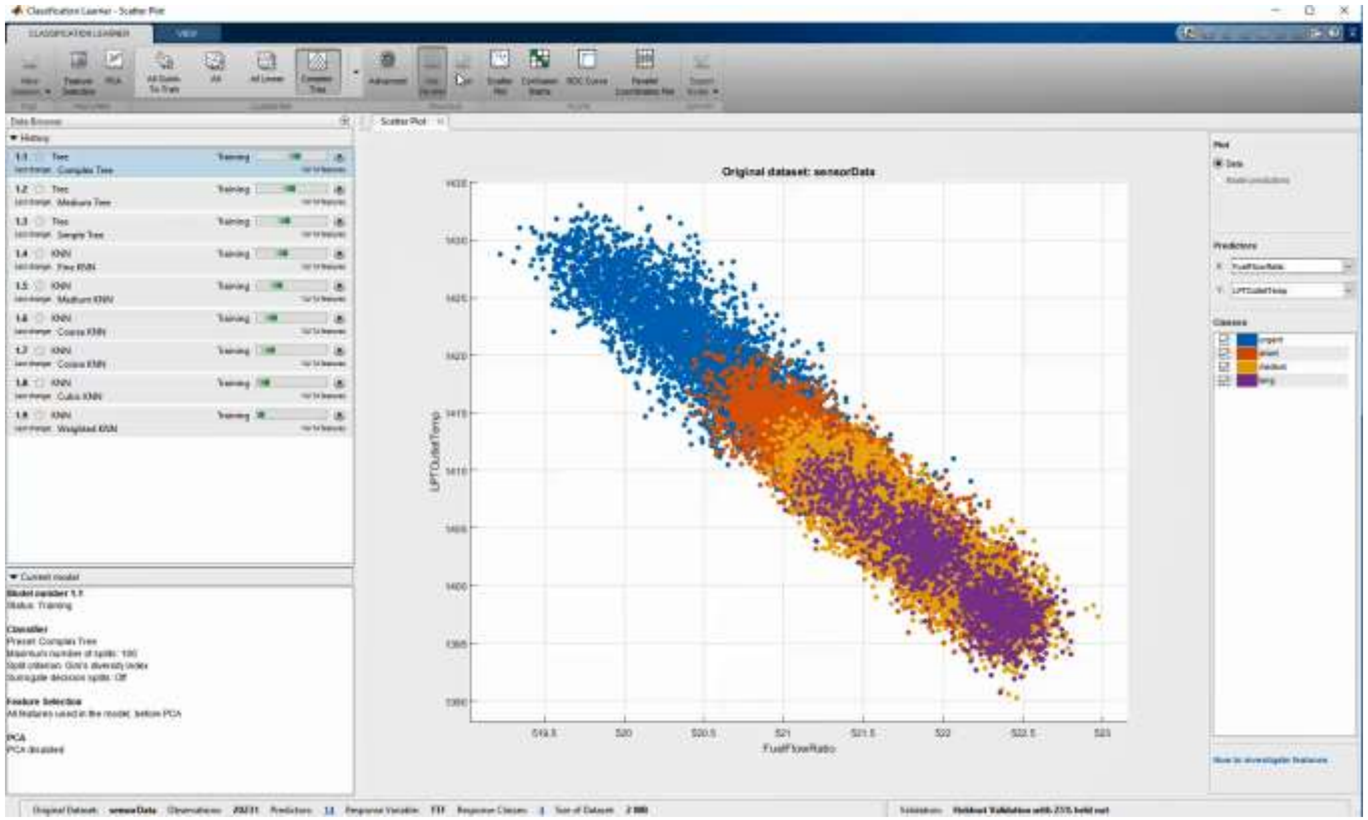
- Linear Model and Stepwise Regression
  - `fitlm, stepwiselm`

- Two month In-Sample and one month Out-Sample
  - `timerange`

Machine Learning App

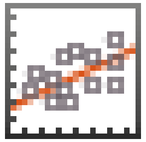
- Point and click interface – no coding required
- Quickly evaluate, compare and select regression models
- Export and share MATLAB code or trained models

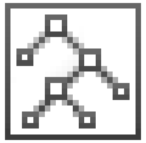




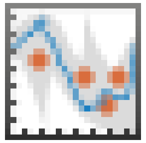
Regression Learner App

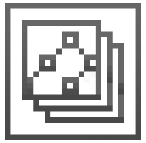
Same workflow as Classification Learner:

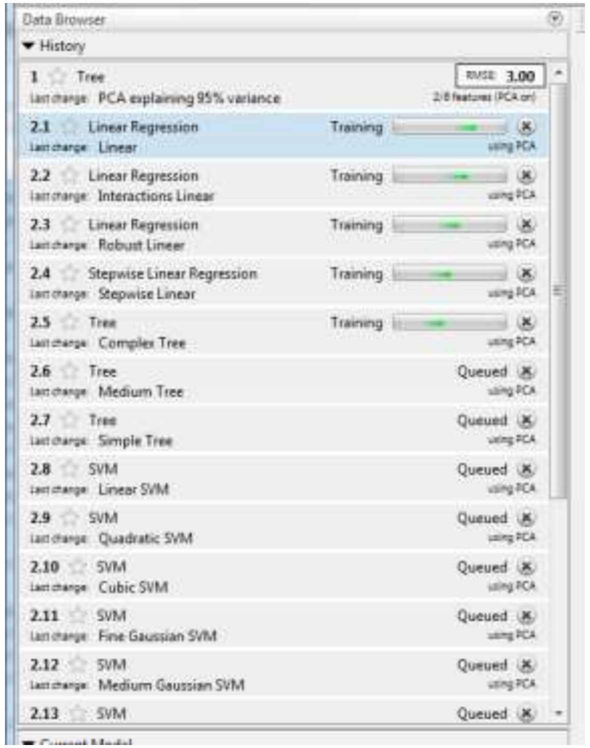

Linear Regression


Trees


SVMs


Gaussian Process Regression


Ensembles



Demo – Long, Short Classification

- Repeat this process, switching to machine learning and **supervised learning**

- Supervised learning
  - *The machine learning task of inferring a function from **labelled** training data*

- Our labelling
  - Look at the median bid/offer spread (in pips)
  - Classify our problem as
    - +1 - where the future return is +ive & greater than the spread (i.e. go long)
    - -1 - where the future return is -ive & greater than the spread (go short)
    - 0 – all other cases

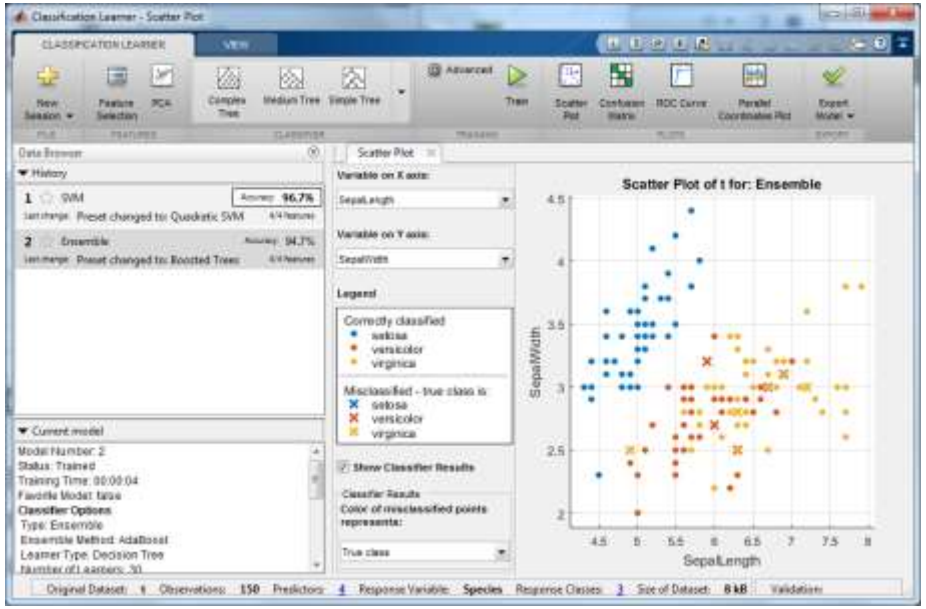
Building the classification model

- Form the predictor/response table
  - Yesterday’s factor row, today’s return

F1	F2	F3	F4	F5	…	FN	R
F1(1)	F2(1)	F3(1)	F4(1)	F5(1)	…	FN(1)	R(2)
F1(2)	F2(2)	F3(2)	F4(2)	F5(2)	…	FN(2)	R(3)
F1(3)	F2(3)	F3(3)	F4(3)	F5(3)	…	FN(3)	R(4)
…	…	…	…	…	…	…	…
F1(M-1)	F2(M-1)	F3(M-1)	F4(M-1)	F5(M-1)	…	FN(M-1)	R(M)

Classification Learner App

- App to apply advanced classification methods to your data
  - Discriminant analysis
  - Dimension reduction via PCA
  - Parallel coordinates plot
  - Categorical predictors
  - Train classifiers in parallel

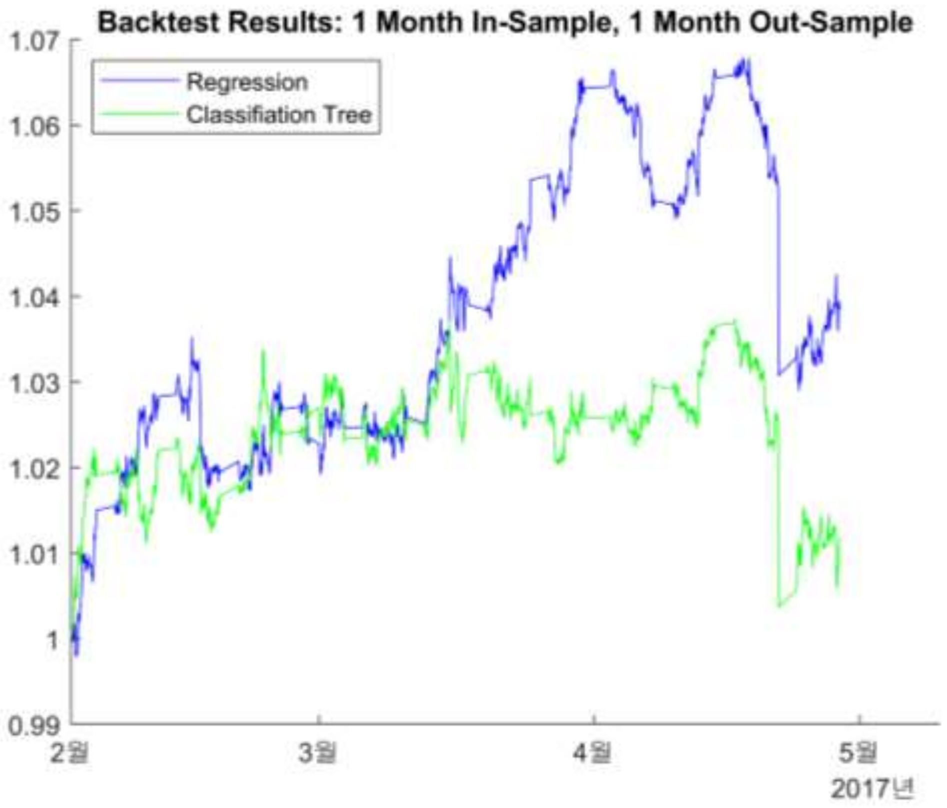


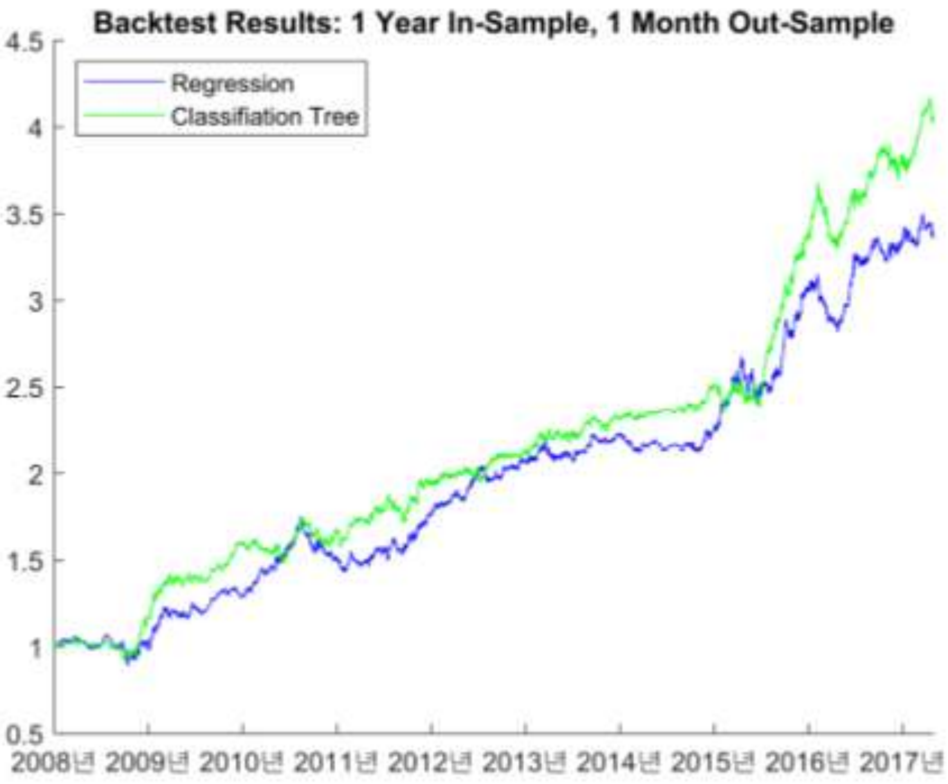
- Also: Table and categorical support via command line

Demo – Historical Backtesting

- Use MATLAB scripting as a backtesting environment
- Loop through our dataset using `datetime` and `dateshift`
  - Run `fitlm` and `fittree` at each iteration
- Adding transaction costs where we trade
  - Once again, based on the bid/ask spread

BackTest Result





Feature Selection

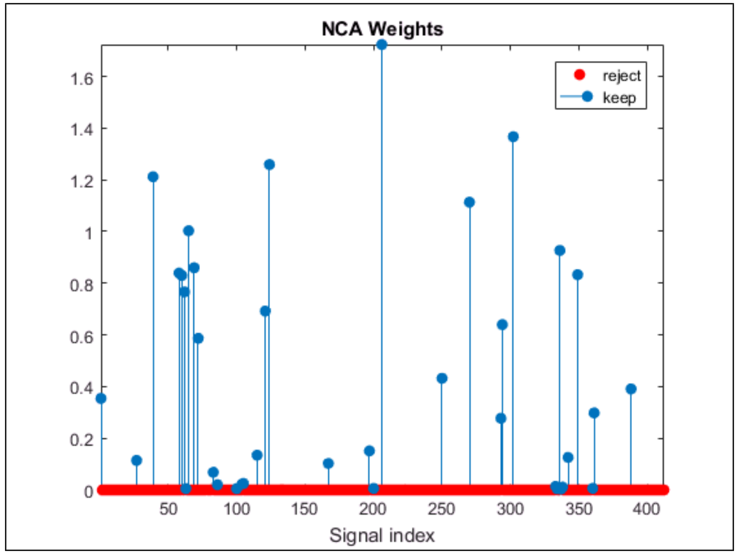
Why?
- Reduce data size (compute/storage gains) and model complexity (prevent overfitting)

When?
- High dimensional datasets with poor feature to observation ratio

Capabilities
- Accuracy comparable to state-of-art techniques
- Regularization to control sparsity and redundancy
- Handles high dimensional data and scales to large datasets

Feature Selection with Neighborhood Component Analysis

X1	X2	X3	X4	X5	X6	X7	Y
3030.9	2564	2187.7	1411.1	1.3602	100	97.613	'pass'
3095.8	2465.1	2230.4	1463.7	0.8294	100	102.34	'pass'
2932.6	2559.9	2186.4	1698	1.5102	100	95.488	'fail'
2988.7	2479.9	2199	909.79	1.3204	100	104.24	'pass'
3032.2	2502.9	2233.4	1326.5	1.5334	100	100.4	'pass'
2946.3	2432.8	2233.4	1326.5	1.5334	100	100.4	'pass'
3030.3	2430.1	2230.4	1463.7	0.8294	100	102.34	'pass'
3058.9	2690.2	2248.9	1004.5	0.7884	100	106.24	'pass'
2967.7	2600.5	2248.9	1004.5	0.7884	100	106.24	'pass'
3016.1	2428.4	2248.9	1004.5	0.7884	100	106.24	'pass'



Fine-tuning Model Parameters

Why?

- Manual parameter selection is tedious and may result in suboptimal performance

When?

- When training a model with one or more parameters that influence the fit

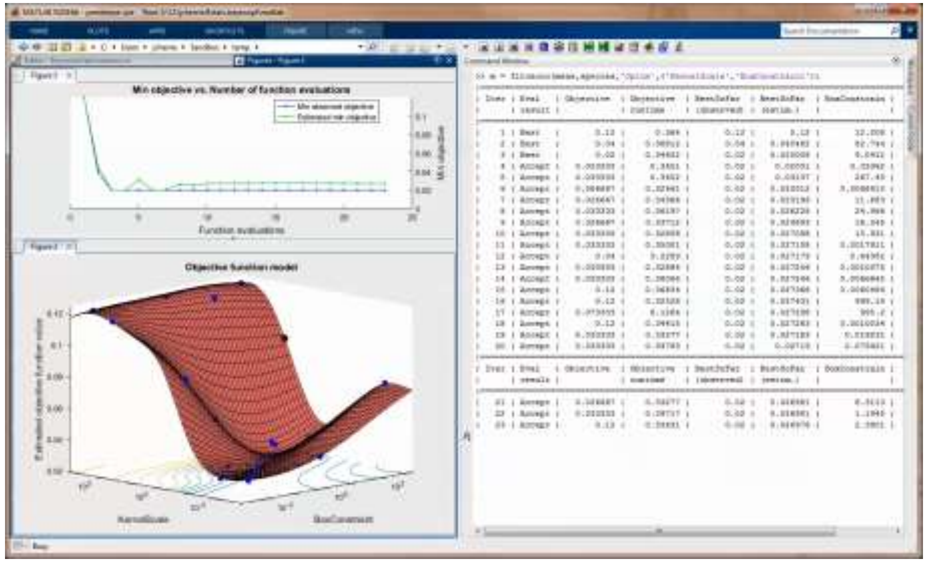
Capabilities

- Efficient comparted to standard optimization techniques or grid search
- Tightly integrated with fit function API with pre-defined optimization problem (e.g. bounds)

Hyperparameter Tuning with Bayesian Optimization

```
template = templateSVM(...
    'KernelFunction', 'linear', ...
    'PolynomialOrder', [], ...
    'KernelScale', 0.25, ...
    'BoxConstraint', 0.1, ...
    'Standardize', true);
m = fitcecoc( T, 'Species', 'Learners', template )
```

Previously tuning these parameters was a manual process



R2016b

Machine Learning with Big Data

Why?
- Learning on larger datasets often leads to better generalization but they don't fit in memory

When?
- Data does not fit in memory
- Data lives remotely on clusters

Capabilities
- Functions for deriving summary statistics and generating visualizations
- Machine learning algorithms for classification, regression and clustering

"Tall" data types and functions for out-of-memory data

Tall Data Types

- Text
- Spreadsheet (Excel)
- Database (SQL)
- Image

- table
- cell
- numeric
- cellstr & string
- Date & Time
- categorical

Exploration & Pre-processing

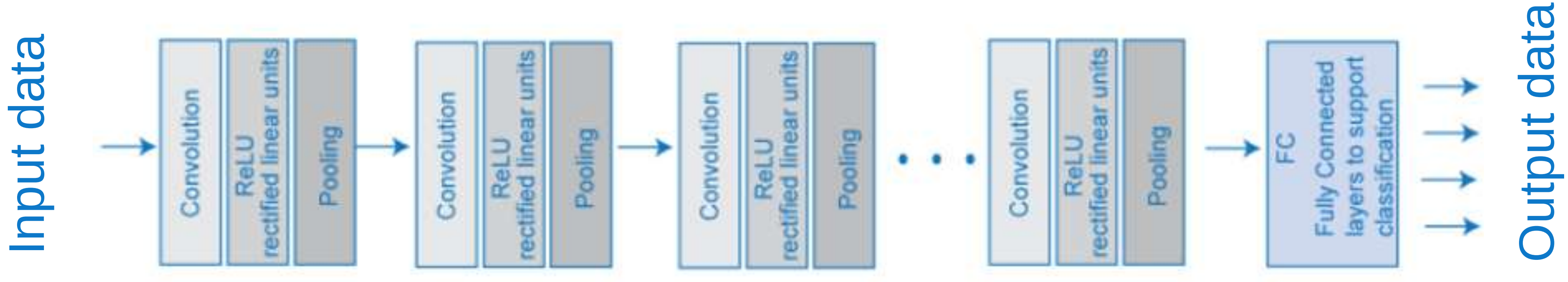
- Numeric functions
- Basic stats reductions
- Date/Time capabilities
- Categorical
- String processing
- Table wrangling
- Missing Data handling
- Summary visualizations:
  - Histogram/histogram2
  - Kernel density plot
  - Bin-scatter

Machine Learning

- Linear Model
- Logistic Regression
- Discriminant analysis
- K-means
- PCA
- Random data sampling
- Summary statistics

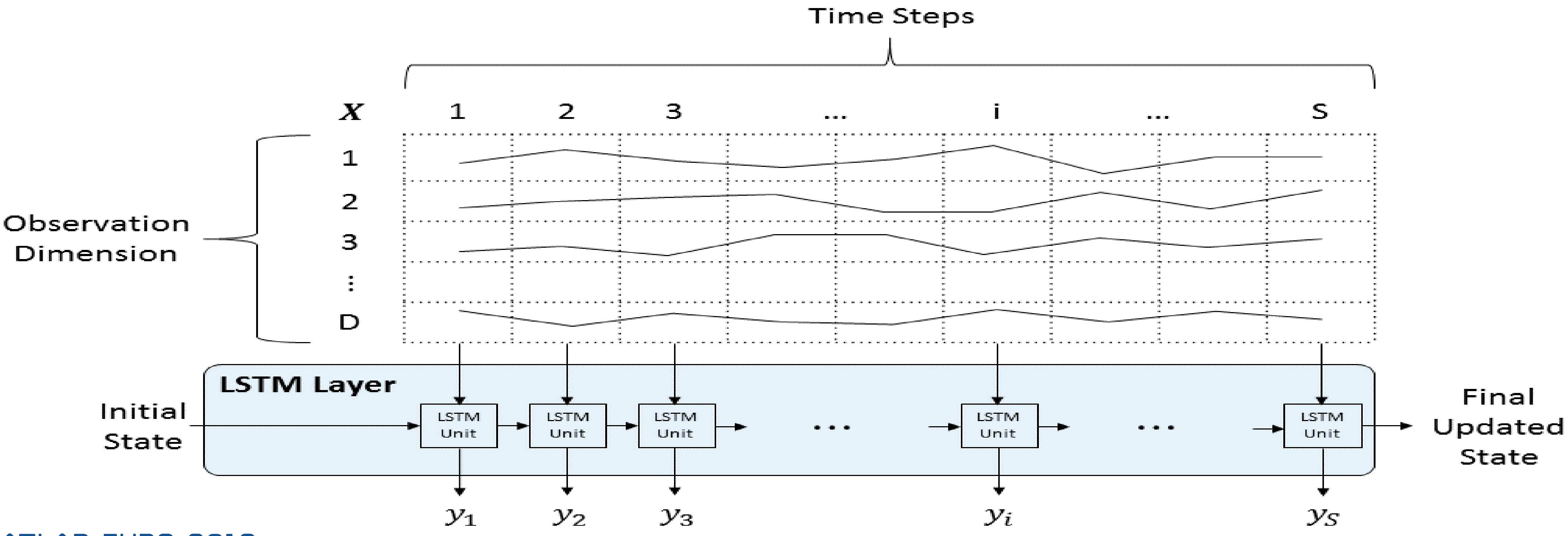
Convolutional Neural Networks (CNN)

- CNN take a fixed size input and generate fixed-size outputs.
- Convolution puts the input images through a set of convolutional filters, each of which activates certain features from the input data.



Time Series Analysis – LSTM Layers

To train a deep neural network to classify sequence data, you can use an LSTM network. An LSTM network enables you to input sequence data into a network, and make predictions based on the individual time steps of the sequence data



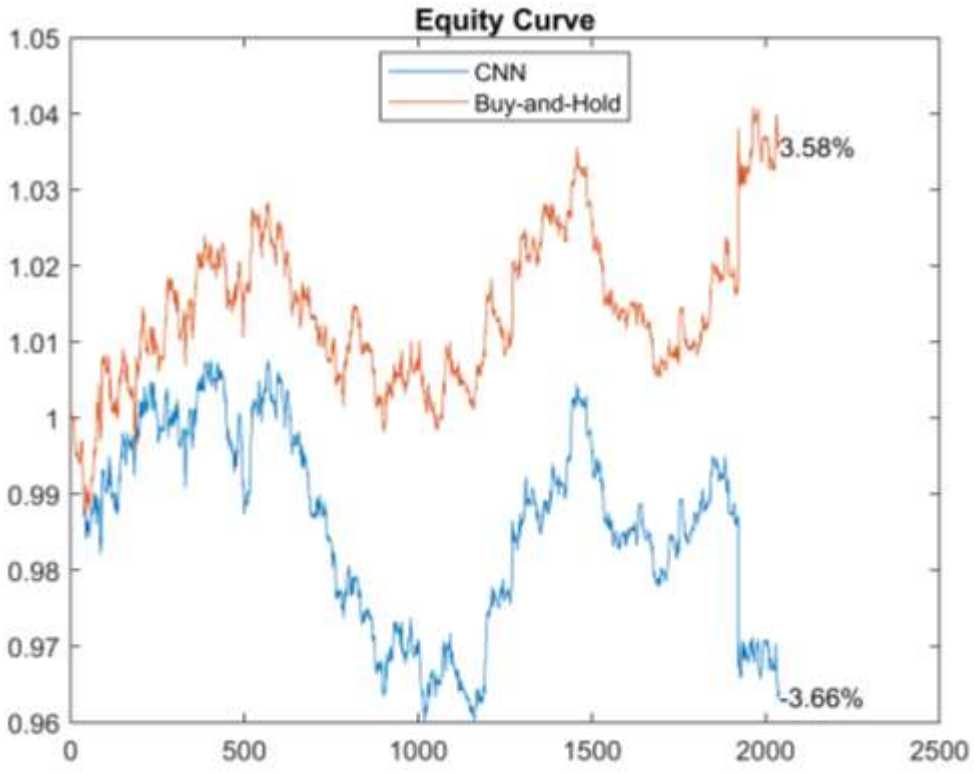
Deep Learning

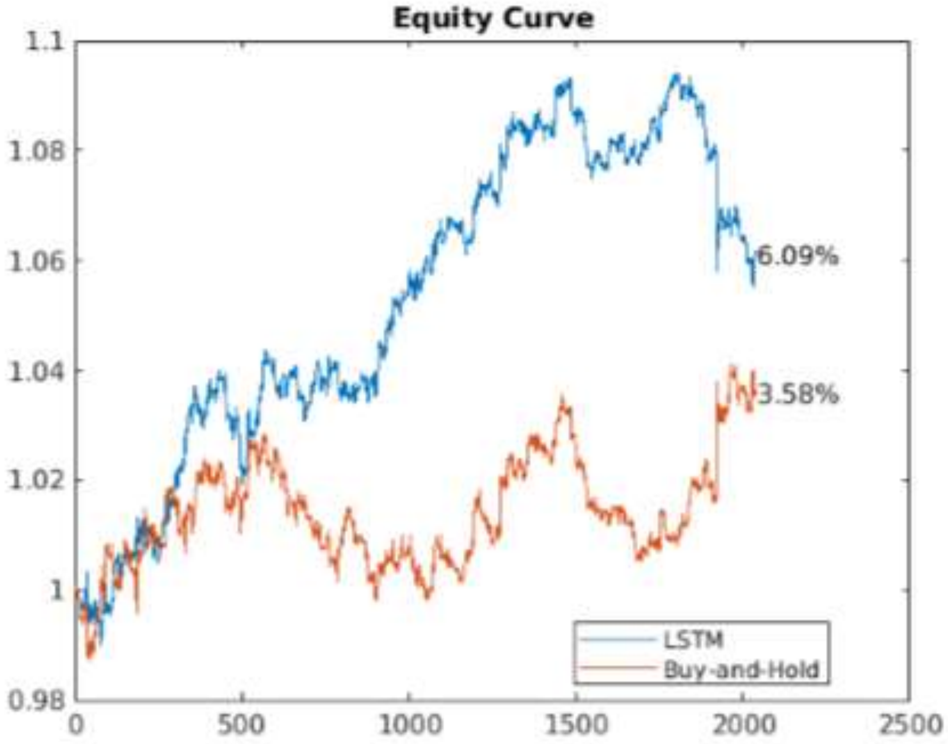
	Classification	Regression

ConvNets	% Define network architecture
	layers = [imageInputLayer([28 28 1])
	convolution2dLayer(5,20)
	fullyConnectedLayer(10)
	softmaxLayer()
	classificationLayer()];
	
	% Train the network.
	options = trainingOptions('adam');
	net = trainNetwork(X, Y, layers, options);
		% Define network architecture
		layers = [imageInputLayer([28 28 1])
		convolutionLayer(5,20)
		fullyConnectedLayer(1)
		regressionLayer()];
		
		% Train the network.
		options = trainingOptions('adam');
		net = trainNetwork(X, Y, layers, options);

LSTM Networks	% Define network architecture
	layers = [sequenceInputLayer(25)
	lstmLayer(100)
	fullyConnectedLayer(10)
	softmaxLayer()
	classificationLayer()];
	
	% Train the network.
	options = trainingOptions('adam');
	net = trainNetwork(X, Y, layers, options);
		% Define network architecture
		layers = [sequenceInputLayer(25)
		lstmLayer(100)
		fullyConnectedLayer(1)
		regressionLayer()];
		
		% Train the network.
		options = trainingOptions('adam');
		net = trainNetwork(X, Y, layers, options);

CNN and LSTM Result





Data Analytics Workflow

Access and Explore Data → **Preprocess Data** → **Develop Predictive Models** → **Integrate Analytics with Systems**

Access and Explore Data
- Files
- Databases
- Sensors

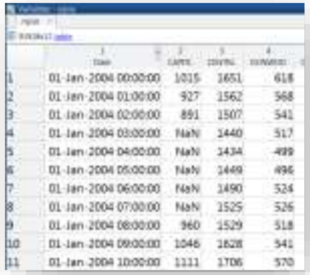
Preprocess Data
- Working with Messy Data
- Data Reduction/ Transformation
- Feature Extraction

Develop Predictive Models
- Model Creation e.g. Machine Learning
- Parameter Optimization
- Model Validation

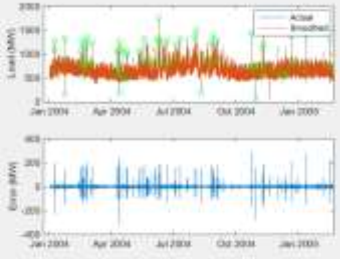
Integrate Analytics with Systems
- Desktop Apps
- Enterprise Scale Systems
- Embedded Devices and Hardware

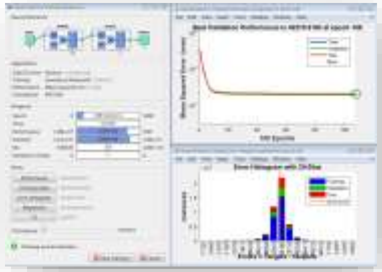
MATLAB Excel .NET .exe C/C++ Java .dll

Solution for Data Analytics



	Time	[illegible]	[illegible]	[illegible]
1	01-Jan-2004 00:00:00	1015	1651	618
2	01-Jan-2004 01:00:00	927	1562	568
3	01-Jan-2004 02:00:00	891	1507	541
4	01-Jan-2004 03:00:00	NaN	1440	517
5	01-Jan-2004 04:00:00	NaN	1434	499
6	01-Jan-2004 05:00:00	NaN	1449	496
7	01-Jan-2004 06:00:00	NaN	1490	524
8	01-Jan-2004 07:00:00	NaN	1525	526
9	01-Jan-2004 08:00:00	960	1529	518
10	01-Jan-2004 09:00:00	1046	1628	541
11	01-Jan-2004 10:00:00	1111	1706	570





Access and Explore Data → **Preprocess Data** → **Develop Predictive Models** → **Integrate Analytics with Systems**

MATLAB Analytics work with business and engineering data

Repositories
- Databases (SQL)
- NoSQL
- Hadoop

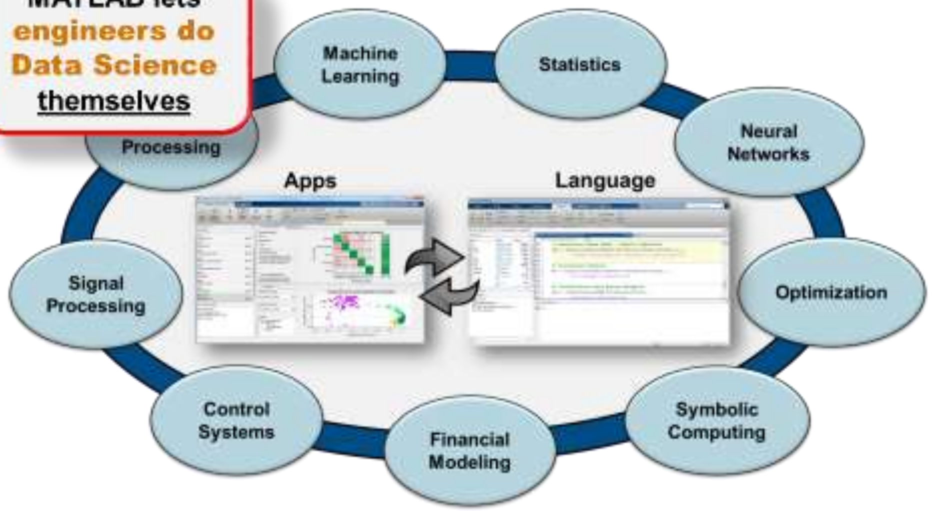
File I/O
- Text
- Spreadsheet
- XML

Web Sources
- RESTful
- JSON
- HTML
- Mapping
- Financial datafeeds

Communication Protocols
- CAN (Controller Area Network)
- DDS (Data Distribution Service)
- OPC (OLE for Process Control)
- XCP (eXplicit Control Protocol)

Real-Time Sources
- Sensors
- GPS
- Instrumentation
- Cameras
- Communication systems
- Machines (embedded systems)

MATLAB lets engineers do Data Science themselves



MATLAB Analytics Scale

- Cloud
- Web
- Cluster
- GPU
- Desktop
- Mobile

MATLAB Analytics run anywhere

Excel Add-in | Hadoop | Standalone Application

C/C++ | Java | .NET

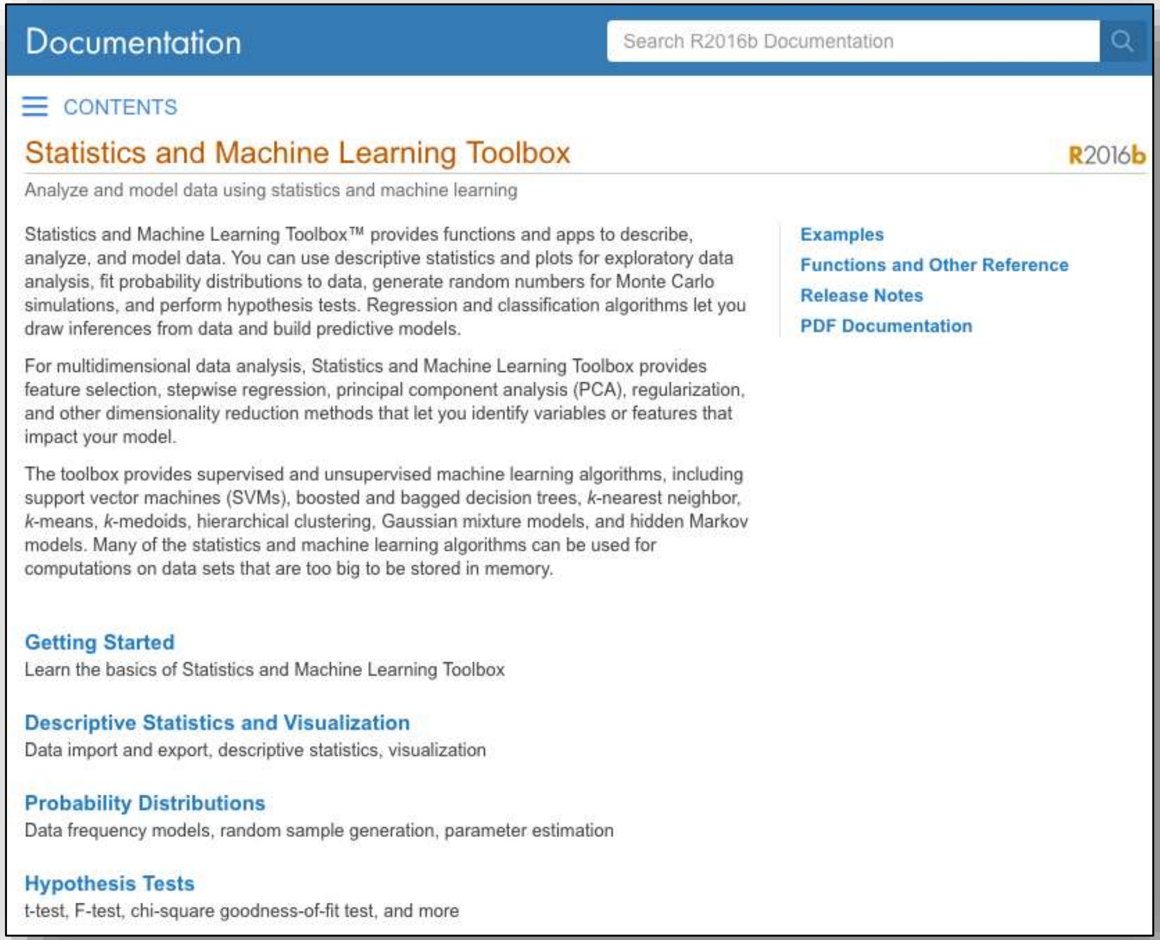
C, C++ | HDL | PLC

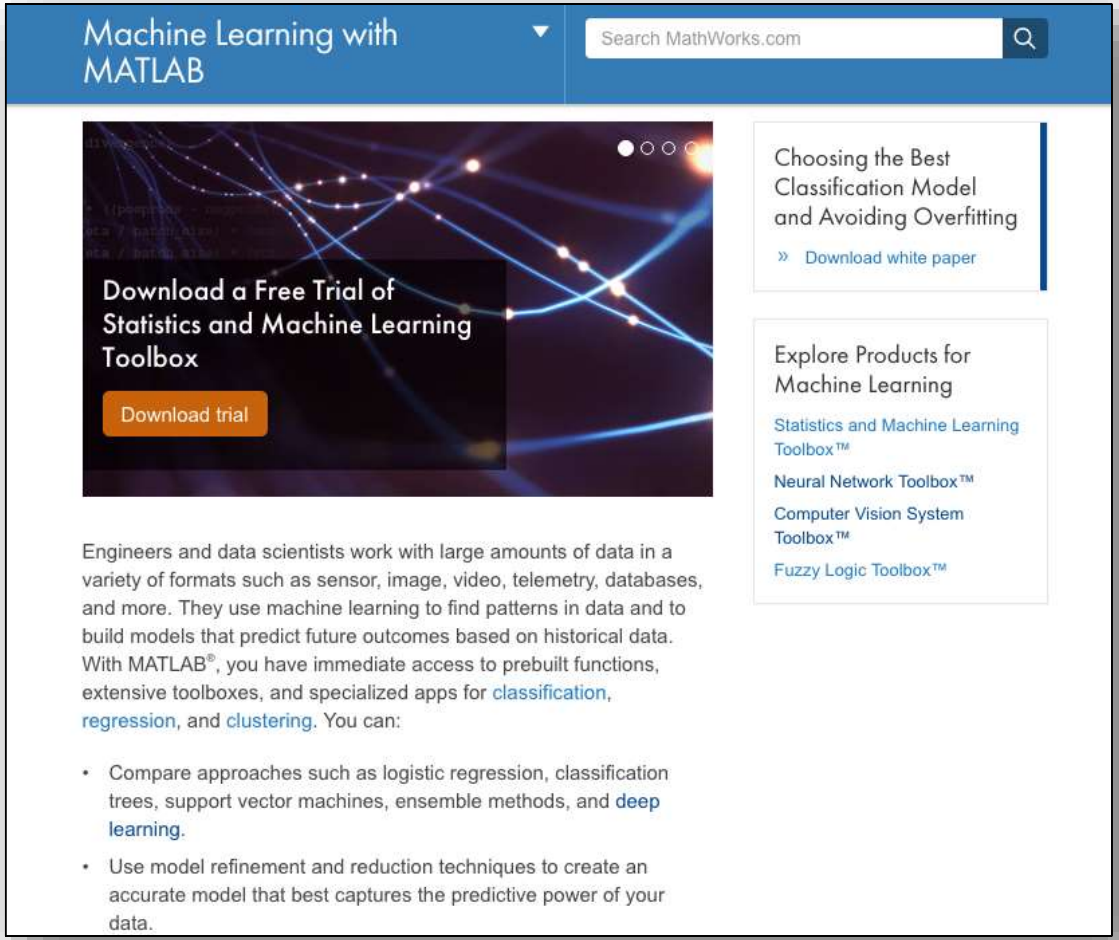
- **Microcontrollers**
- **DSP chips**
- **FPGAs**
- **Industrial automation systems**

Additional Resources

Documentation:

https://www.mathworks.com/solutions/machine-learning.html
https://www.mathworks.com/solutions/deep-learning.html





MathWorks®

Accelerating the pace of engineering and science