

The image features a close-up of two hands, one resting on the other, with a decorative graphic overlay. The graphic consists of several overlapping triangles in shades of blue and orange. The text 'MATLAB EXPO 2018 KOREA' is positioned to the right of the hands.

MATLAB EXPO 2018
KOREA

MATLAB EXPO 2018

MATLAB을 이용한 머신 러닝 (기본)

Senior Application Engineer
엄준상 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

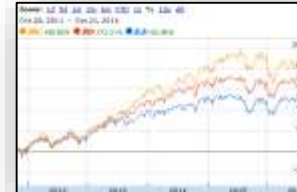
Solution needs to adapt with changing data



Weather Forecasting



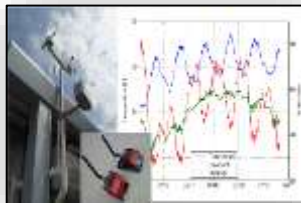
Energy Load Forecasting



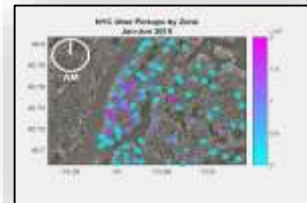
Stock Market Prediction

update as more data becomes available

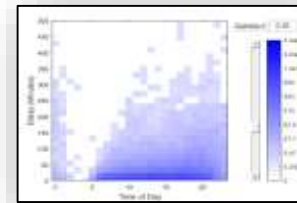
Solution needs to scale



IoT Analytics

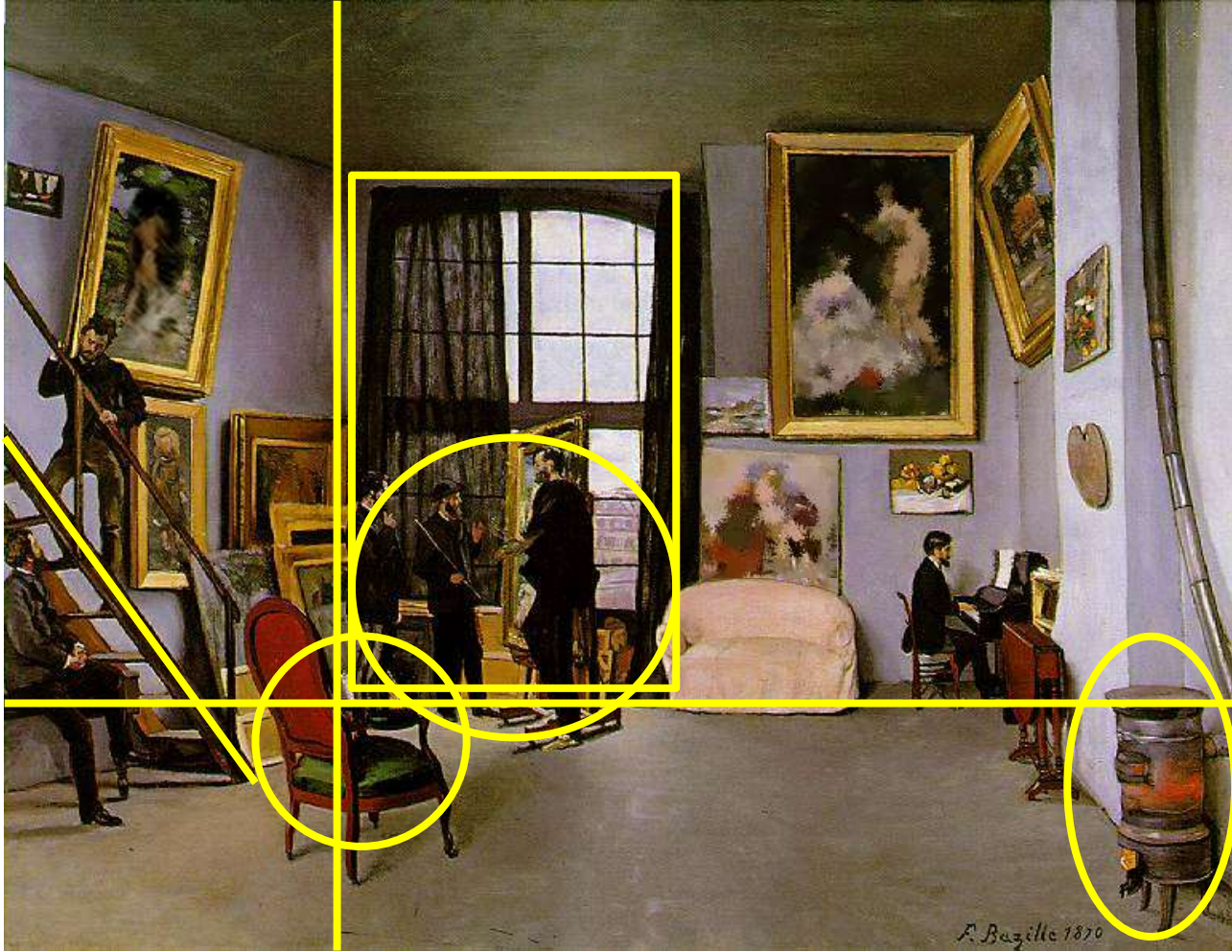


Taxi Availability

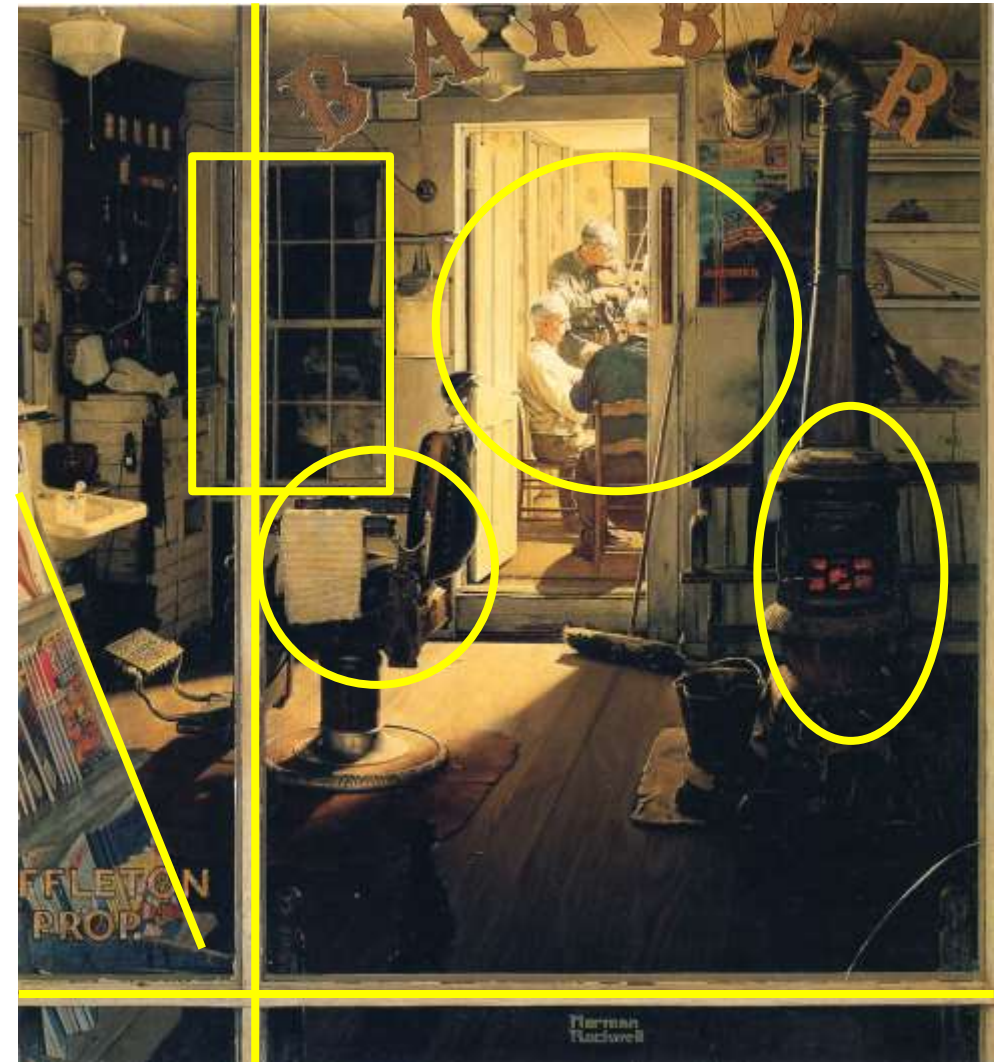


Airline Flight Delays

learn efficiently from very large data sets

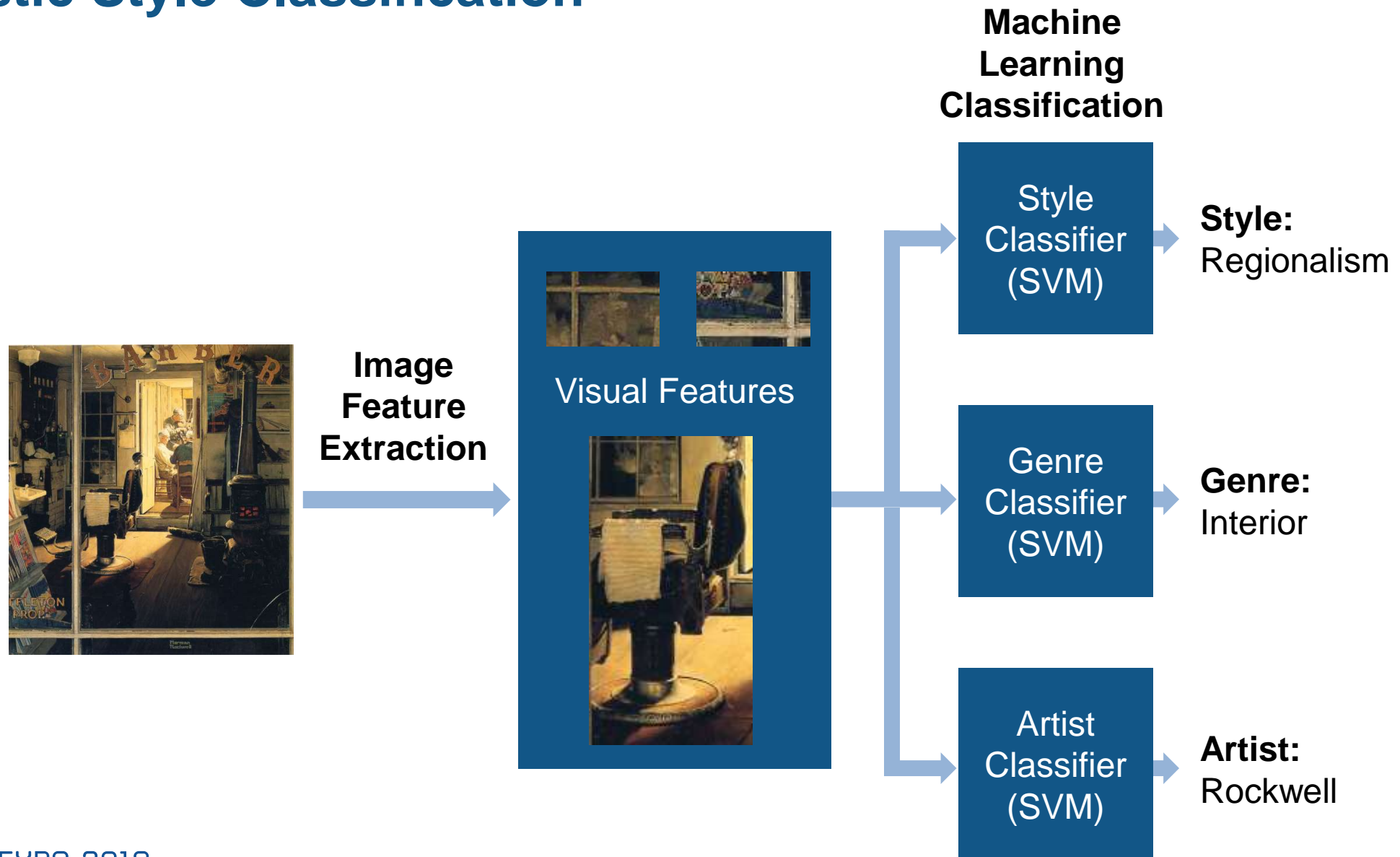


Bazille's Studio
Bazille 1870



Shuffleton's Barbershop
Rockwell 1950

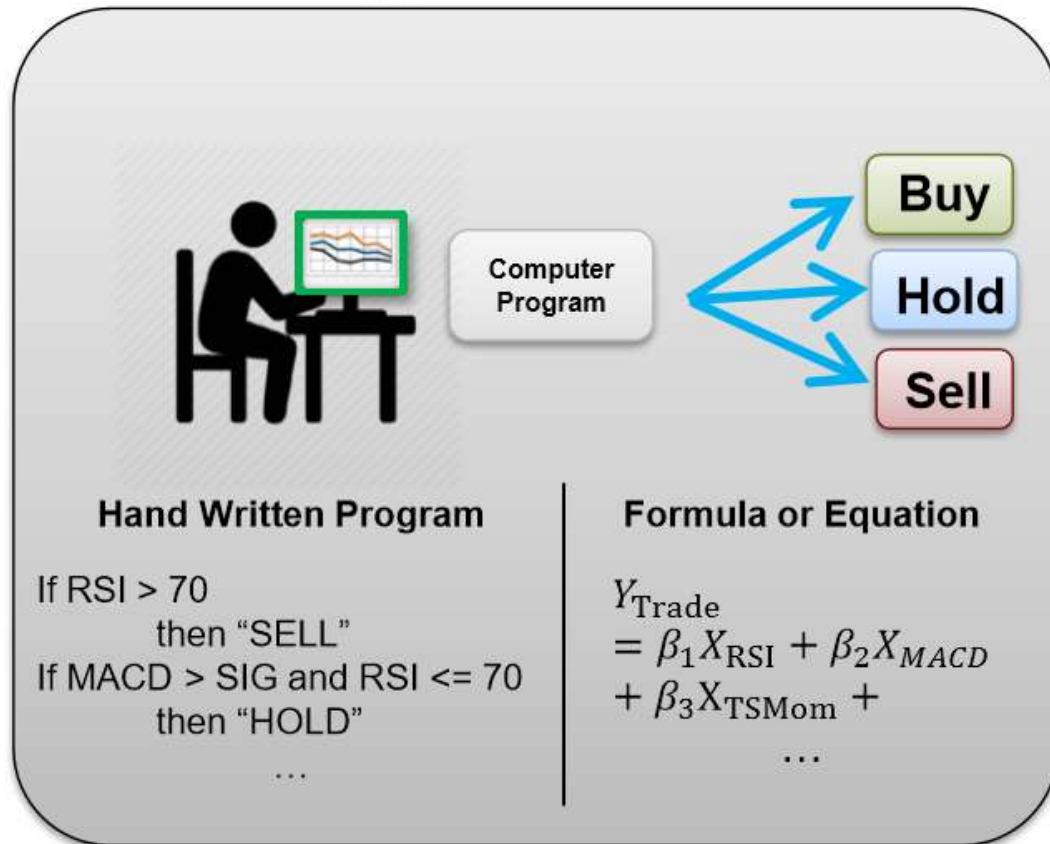
Artistic Style Classification



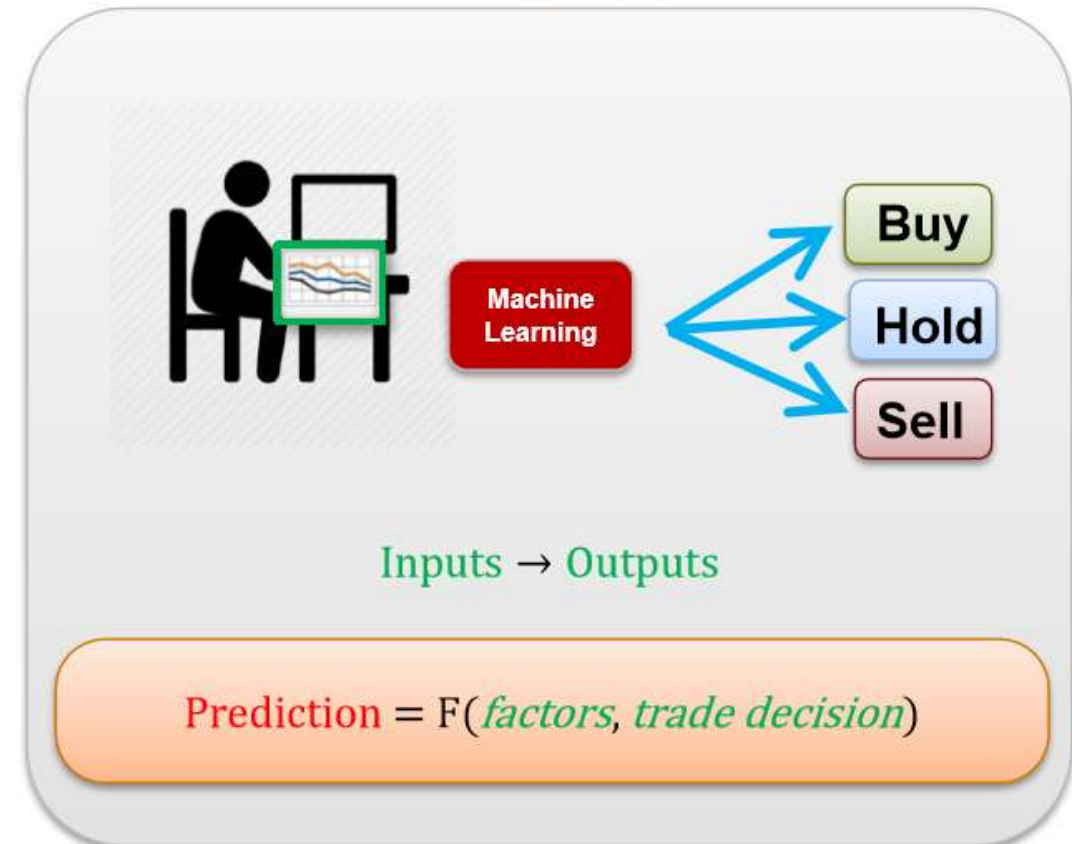
Machine Learning

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

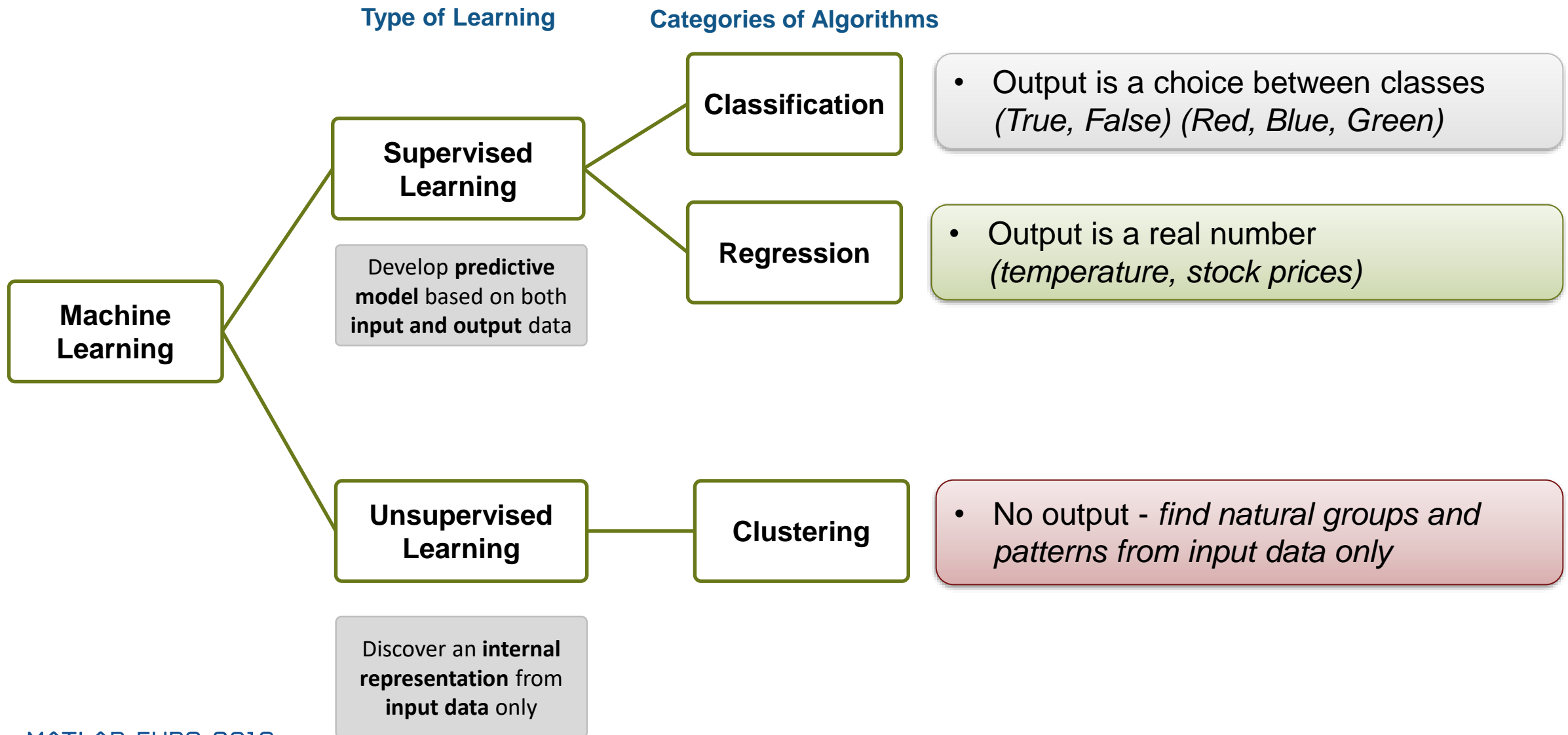
Standard Approach



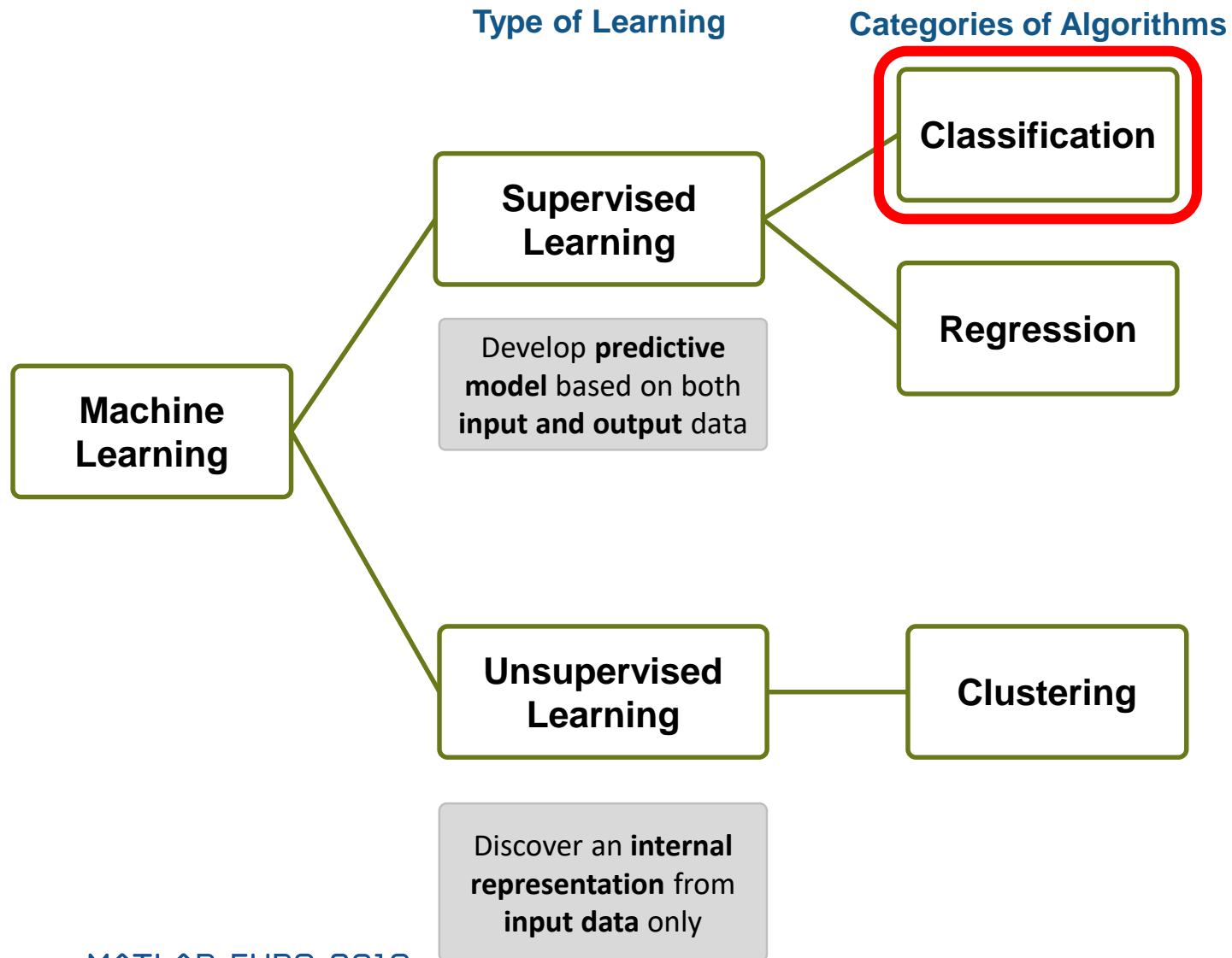
Machine Learning Approach



Different Types of Learning





Example: Classification



Objective:

Train a classifier to classify human activity from sensor data

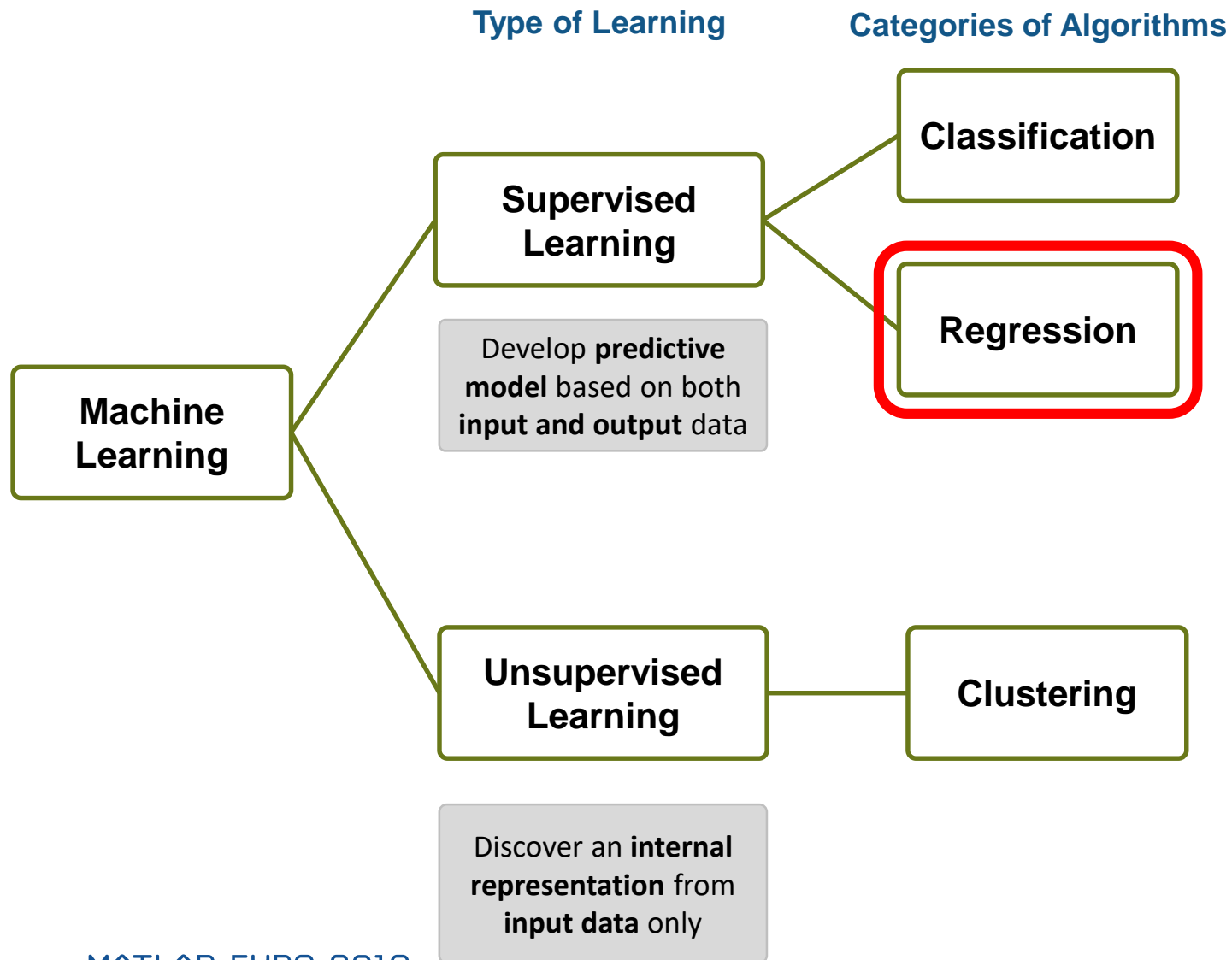
Data:

Inputs	3-axial Accelerometer 3-axial Gyroscope	
Outputs		

Approach:

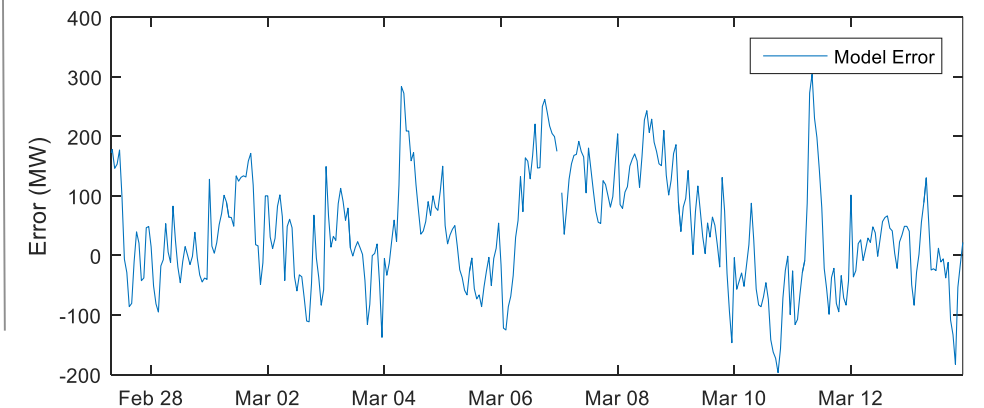
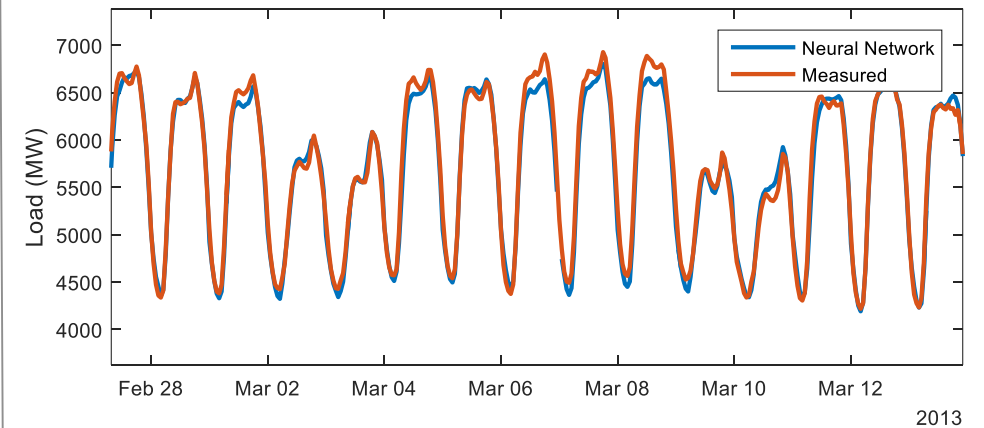
- Import data
- Interactively train and compare classifiers
- Test results on new sensor data

Example: Regression

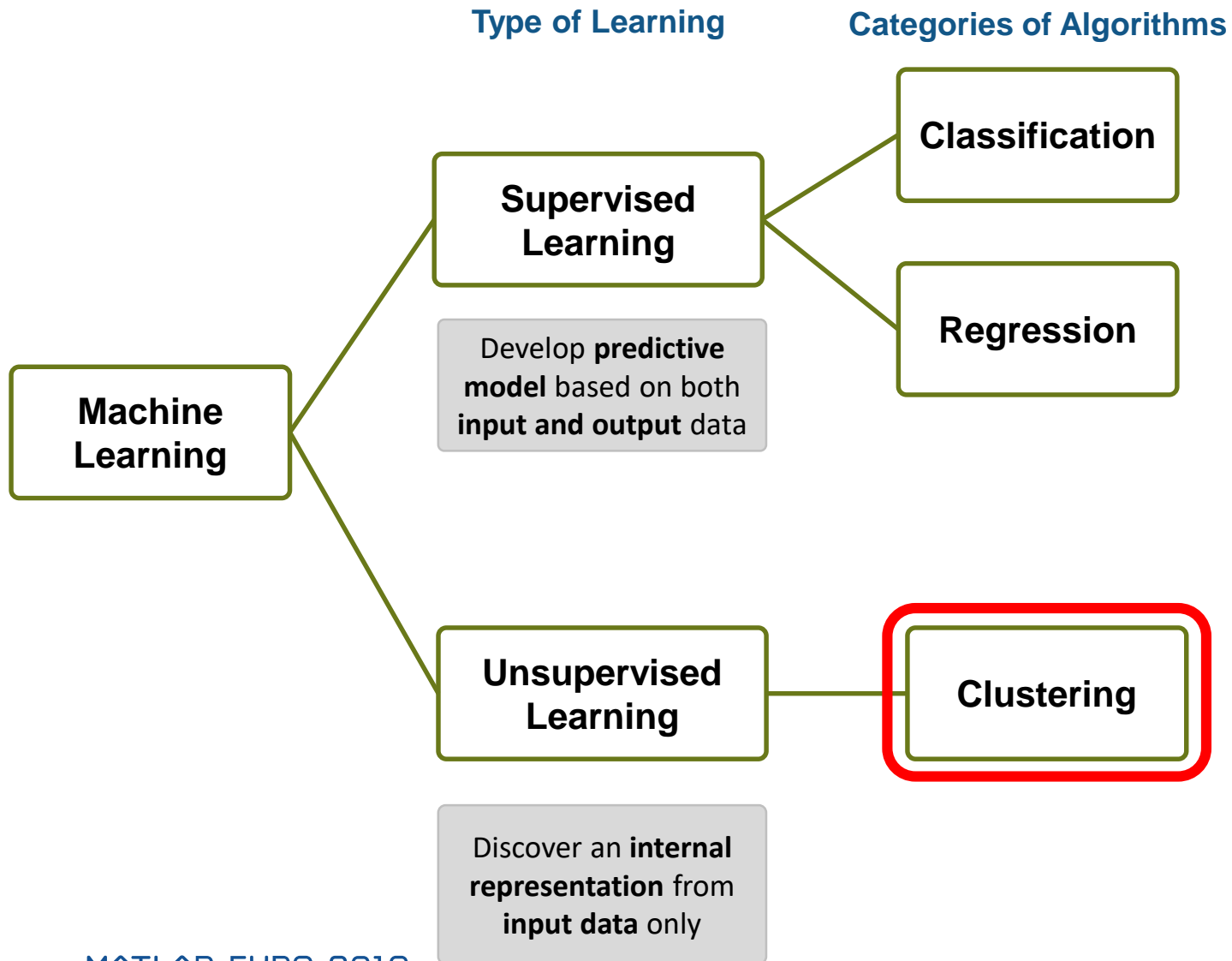


Objective:

Easy and accurate computation of day-ahead system load forecast

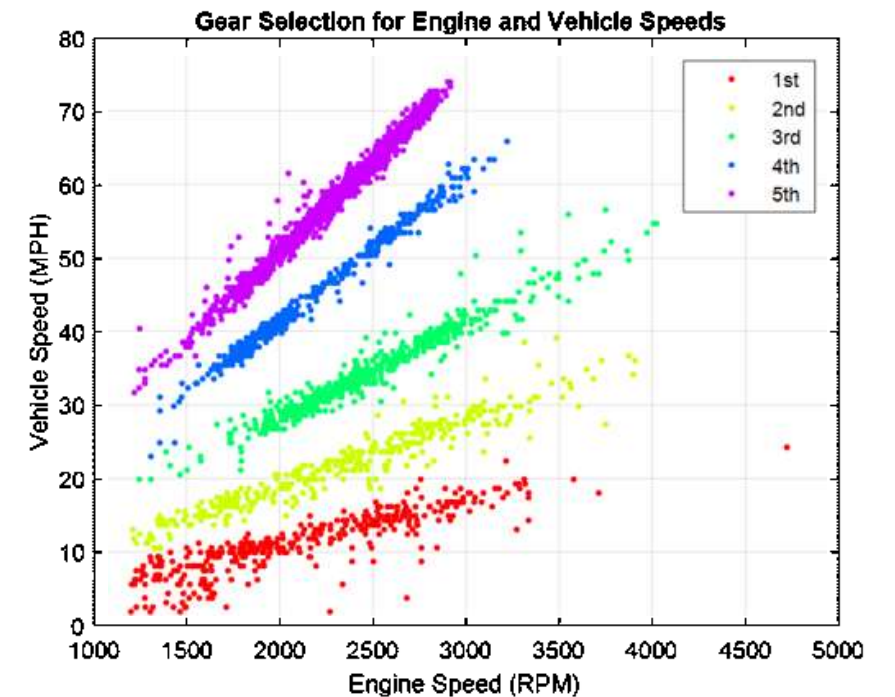


Example: Clustering

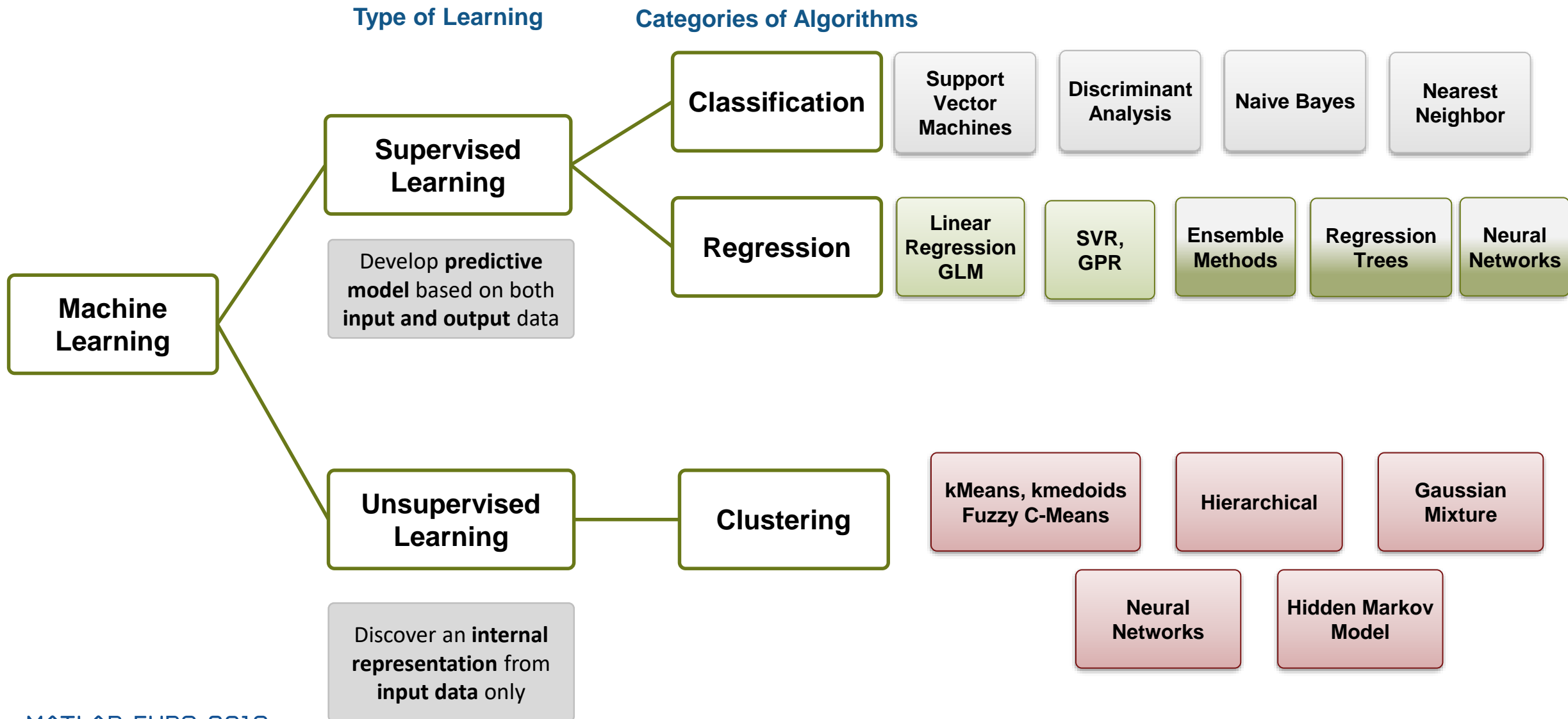


Objective:

Given data for engine speed and vehicle speed, identify clusters



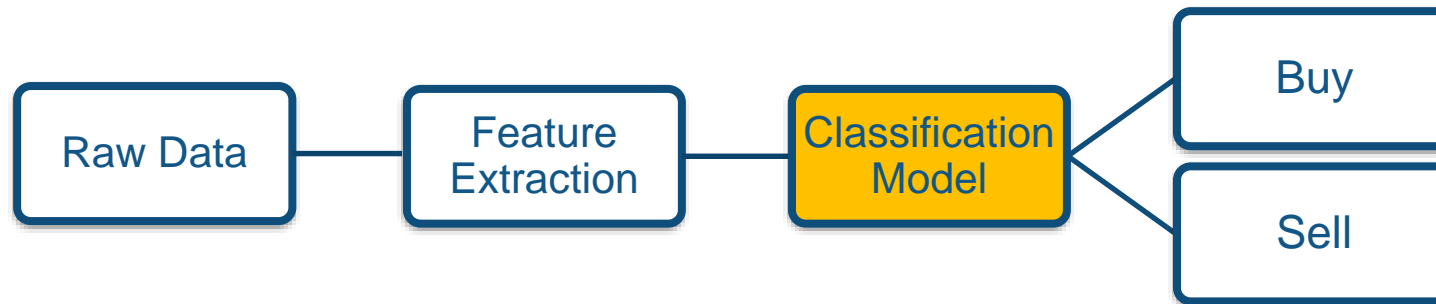
Different Types of Learning



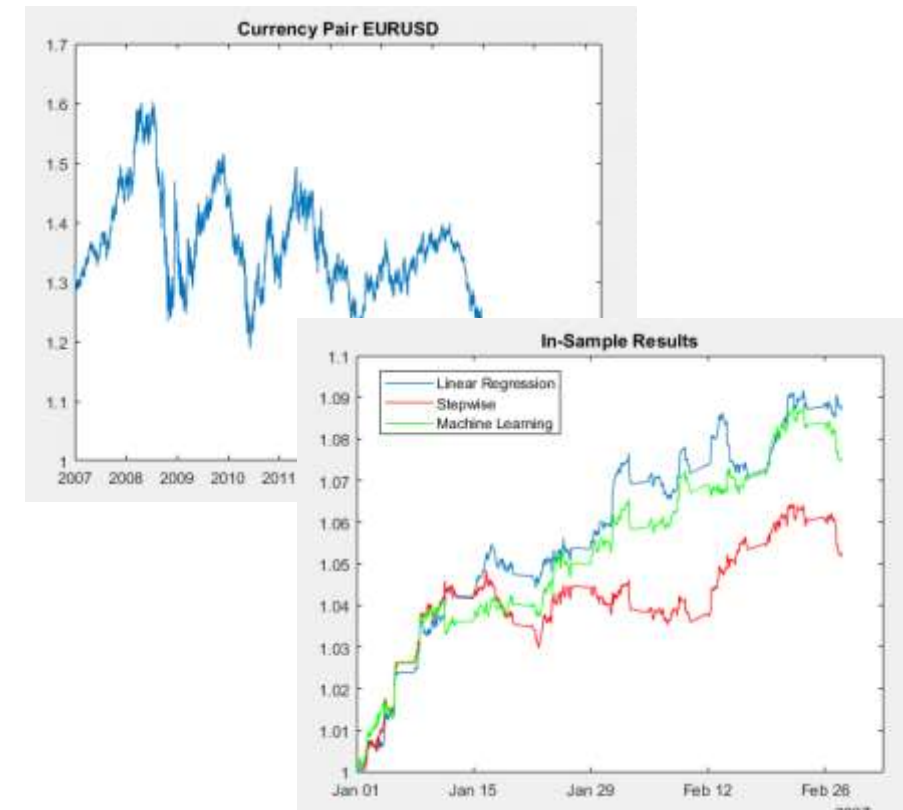
Case Study

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`

<u>Time</u>	<u>Mid</u>	<u>Bid</u>	<u>Ask</u>
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

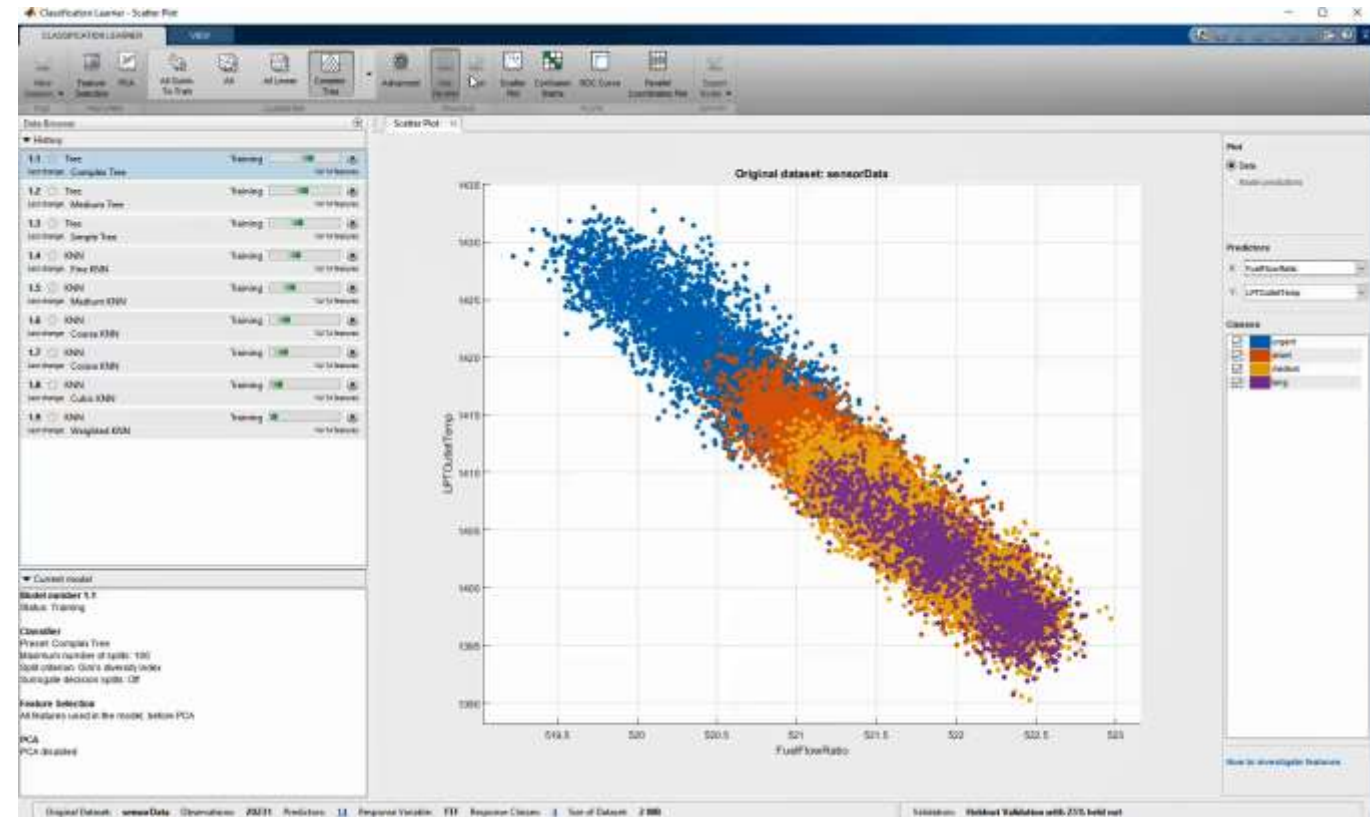
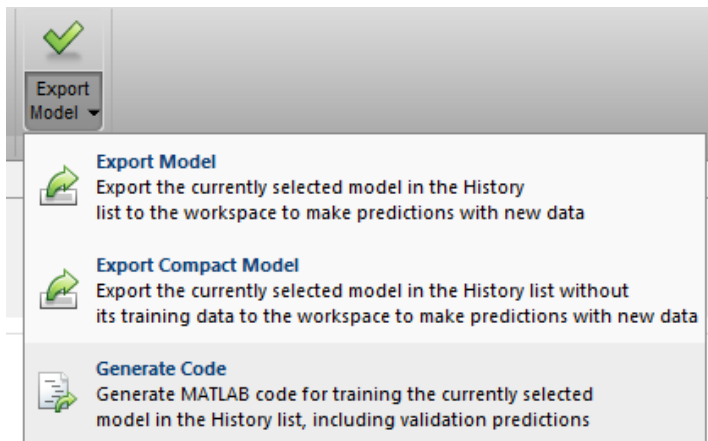
- Train a model based around a number of factors
 - Technical Indicators & Short Term Returns
 - Attempt to predict positive or negative future returns using current information
 - Trade on this prediction
- Model Selection
 - Linear regression and stepwise
 - Classification Tree
- Backtesting
 - Test 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`

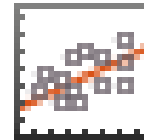
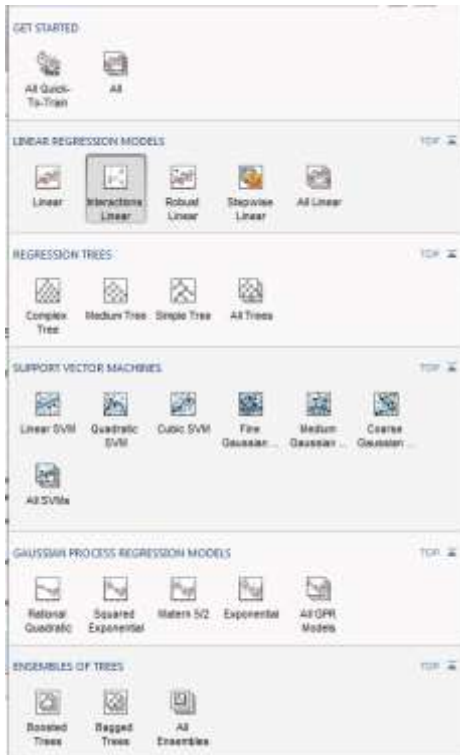
Machine Learning App

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

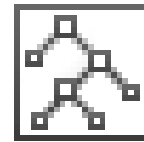


Regression Learner App

Same workflow as Classification Learner:



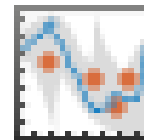
Linear Regression



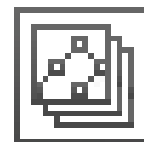
Trees



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

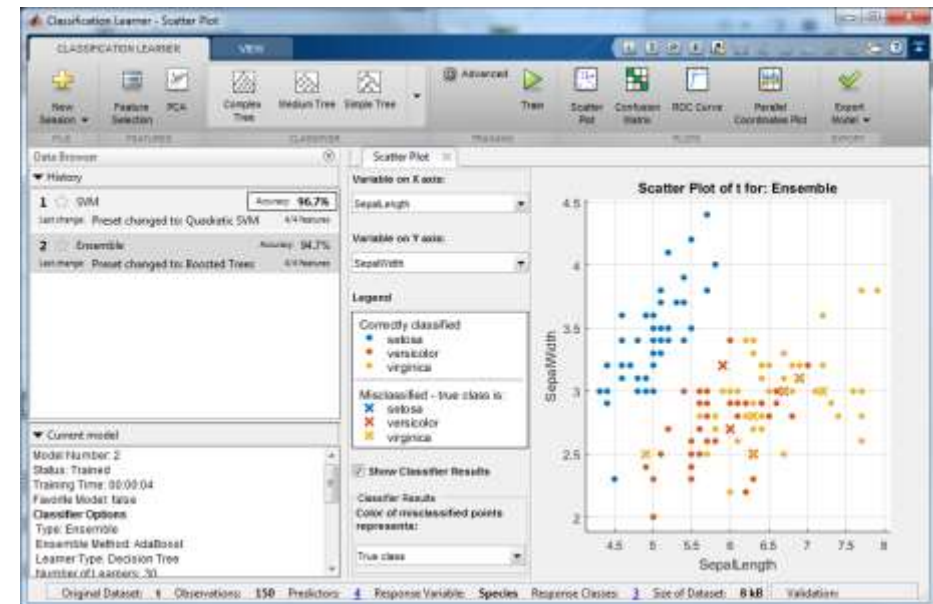
Building the classification model

- 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

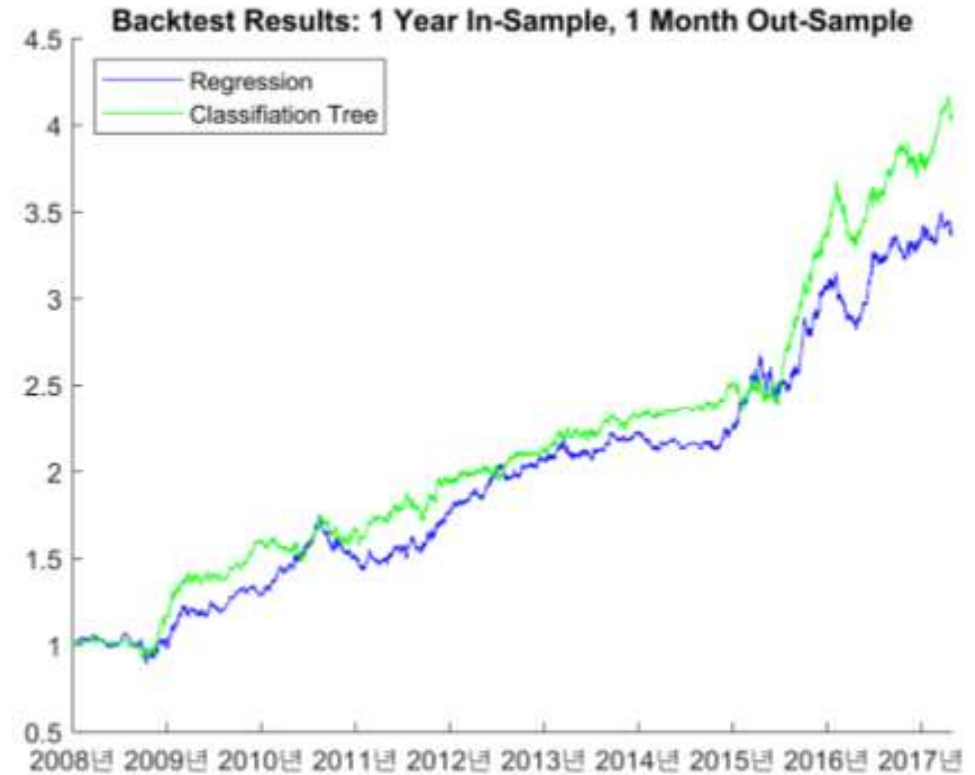
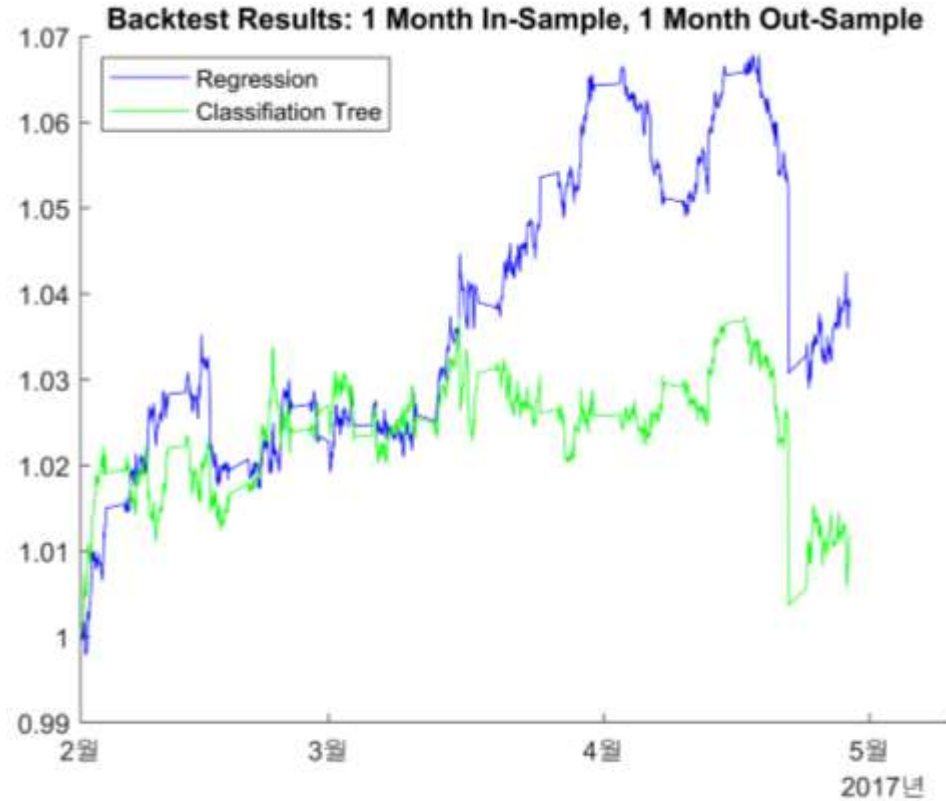


- Also: Table and categorical support via command line

Demo – Historical Backtesting

- Use MATLAB scripting as a backtesting environment
- Loop through our dataset using `datetime` and `dateshift`
 - Run `fitlm` and `fitree` 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?

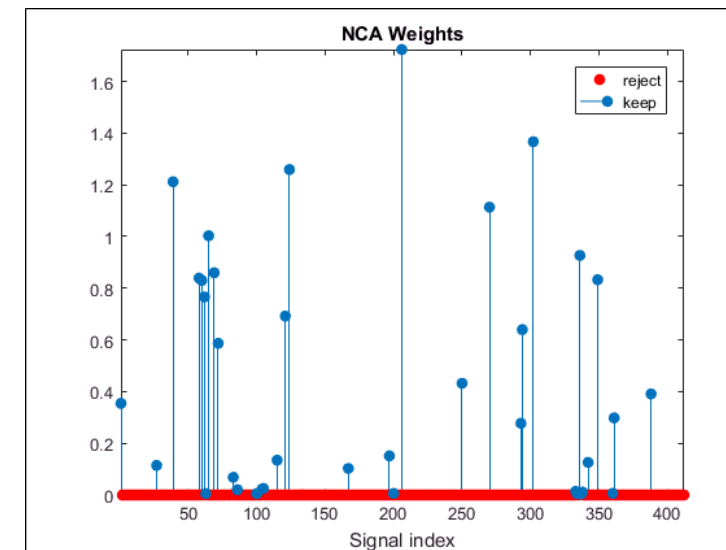
- High dimensional datasets with poor feature to observation ratio

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

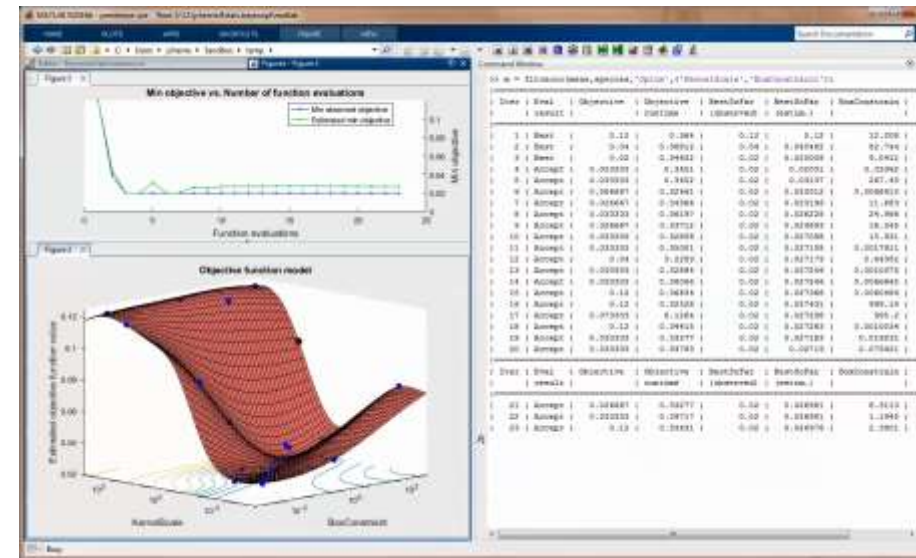
Capabilities

- Efficient compared 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



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

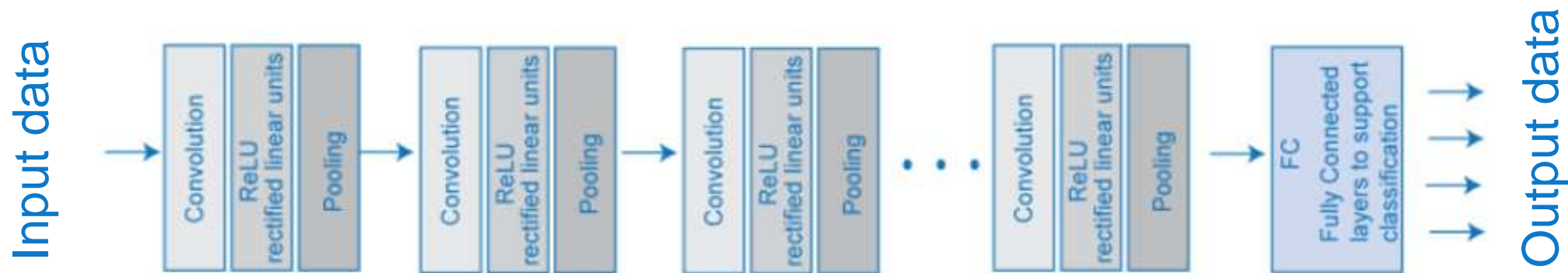
- Numeric functions
- Basic stats reductions
- Date/Time capabilities
- Categorical
- String processing
- Table wrangling
- Missing Data handling
- Summary visualizations:
 - Histogram/histogram2
 - Kernel density plot
 - Bin-scatter

Machine Learning

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

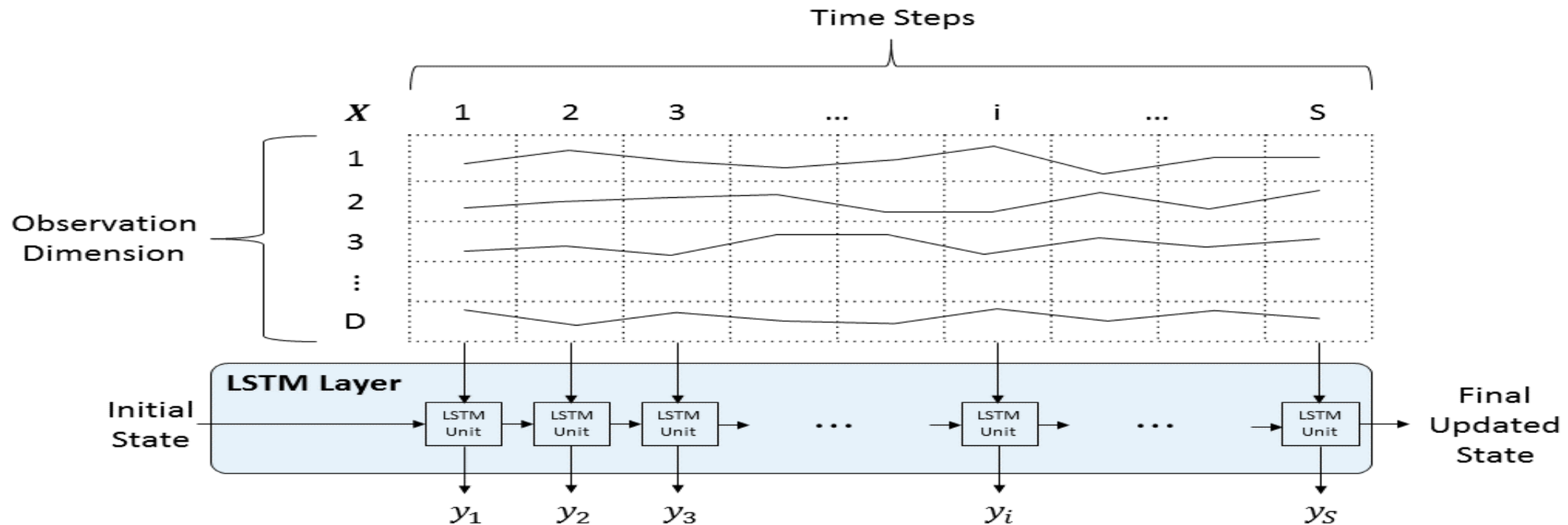
Convolutional Neural Networks (CNN)

- 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

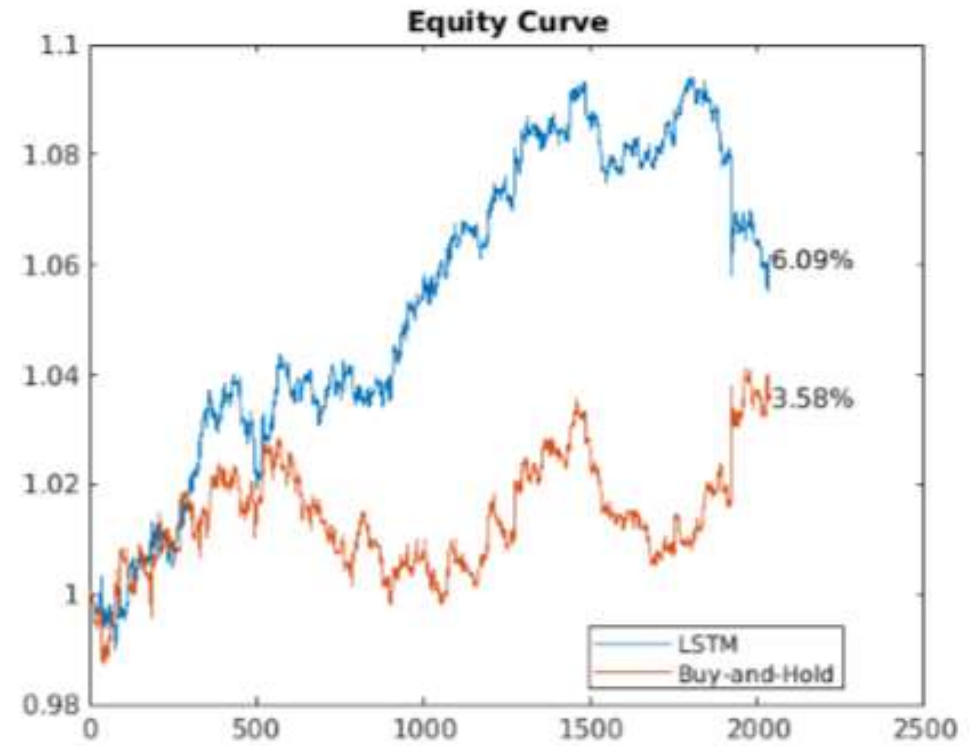
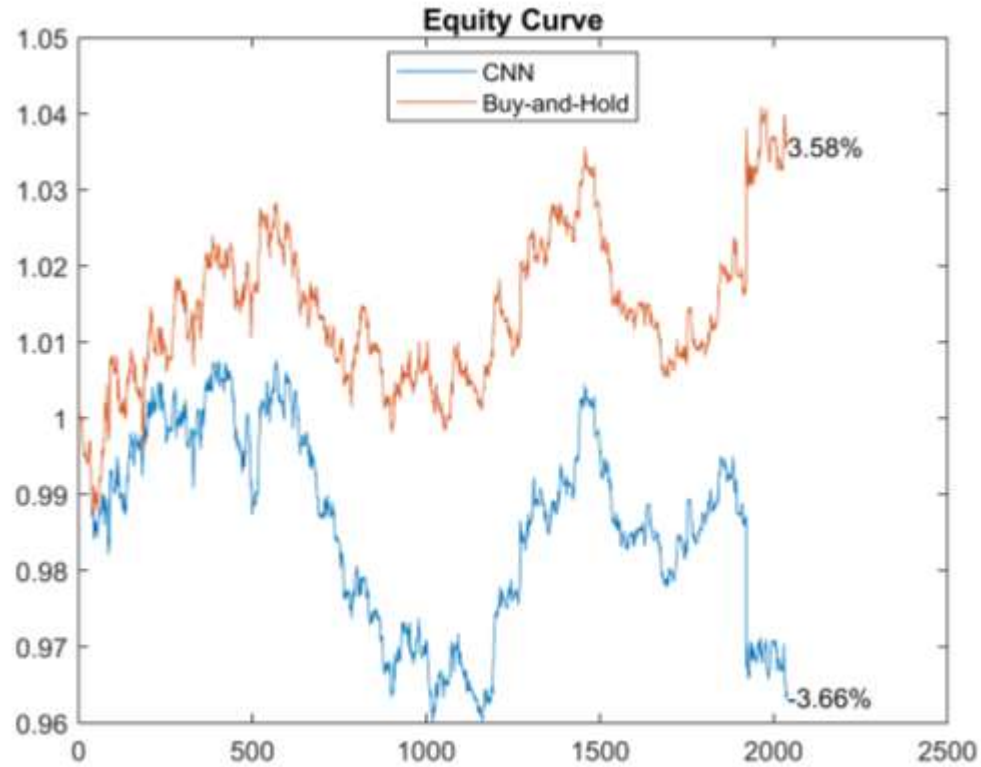
To train a deep neural network to classify sequence data, you can use an LSTM network. An LSTM network enables you to input sequence data into a network, and make predictions based on the individual time steps of the sequence data



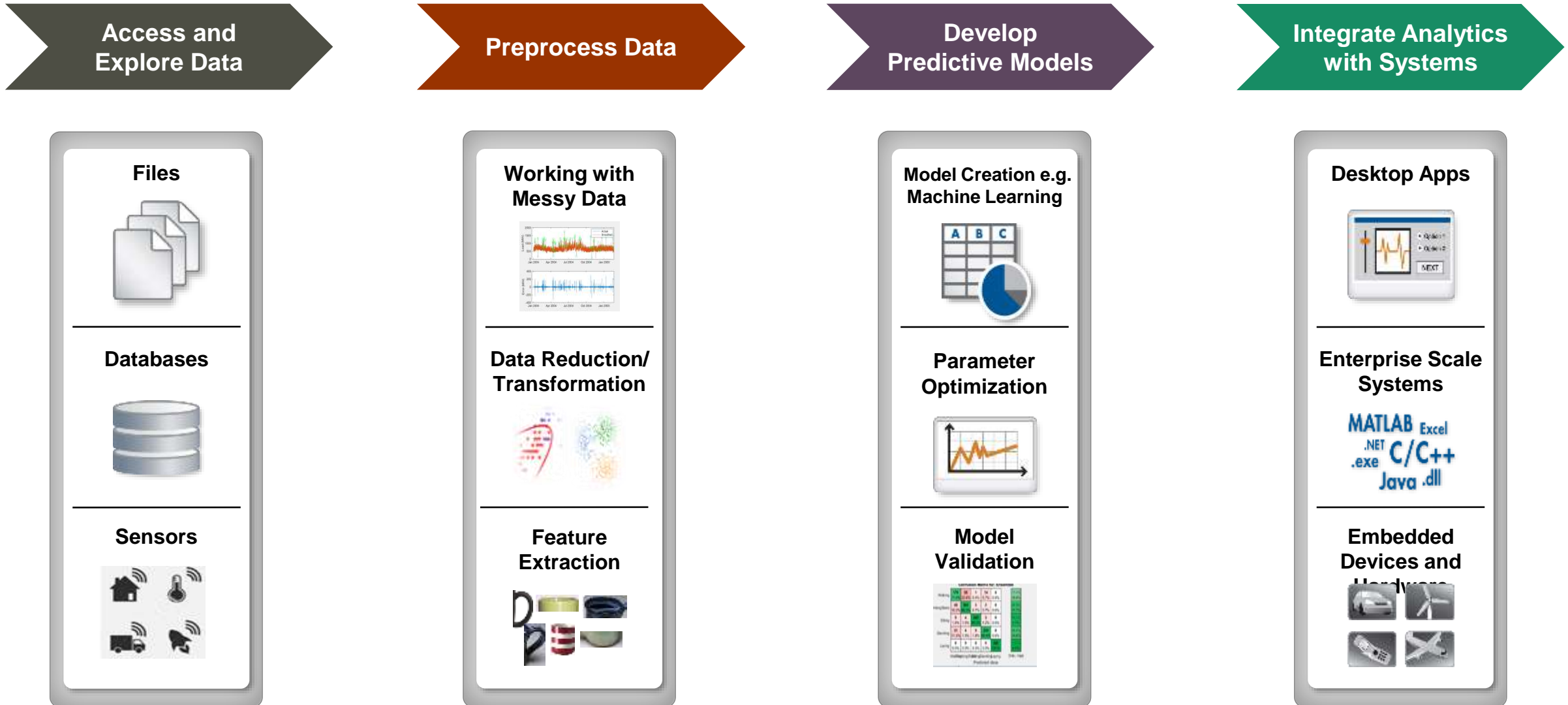
Deep Learning

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

CNN and LSTM Result

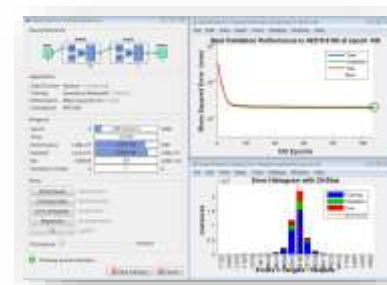
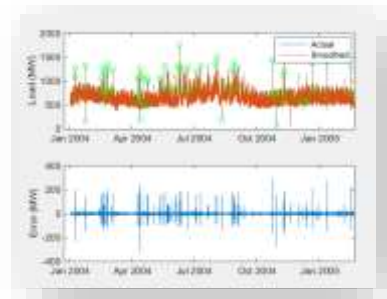


Data Analytics Workflow



Solution for Data Analytics

Time	Temp	Humidity	Pressure
01-Jan-2004 00:00:00	1015	1651	618
01-Jan-2004 01:00:00	927	1562	568
01-Jan-2004 02:00:00	851	1507	541
01-Jan-2004 03:00:00	NaN	1440	517
01-Jan-2004 04:00:00	NaN	1434	499
01-Jan-2004 05:00:00	NaN	1449	496
01-Jan-2004 06:00:00	NaN	1490	524
01-Jan-2004 07:00:00	NaN	1525	526
01-Jan-2004 08:00:00	960	1529	518
01-Jan-2004 09:00:00	1046	1628	541
01-Jan-2004 10:00:00	1111	1706	570



Access and Explore Data

Preprocess Data

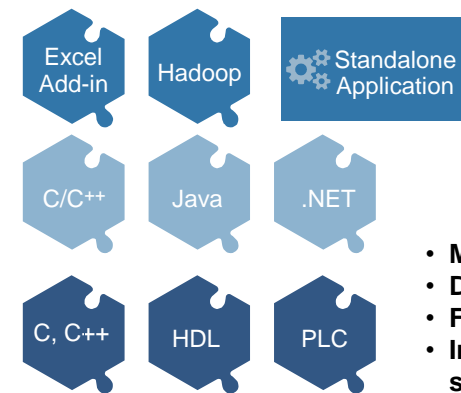
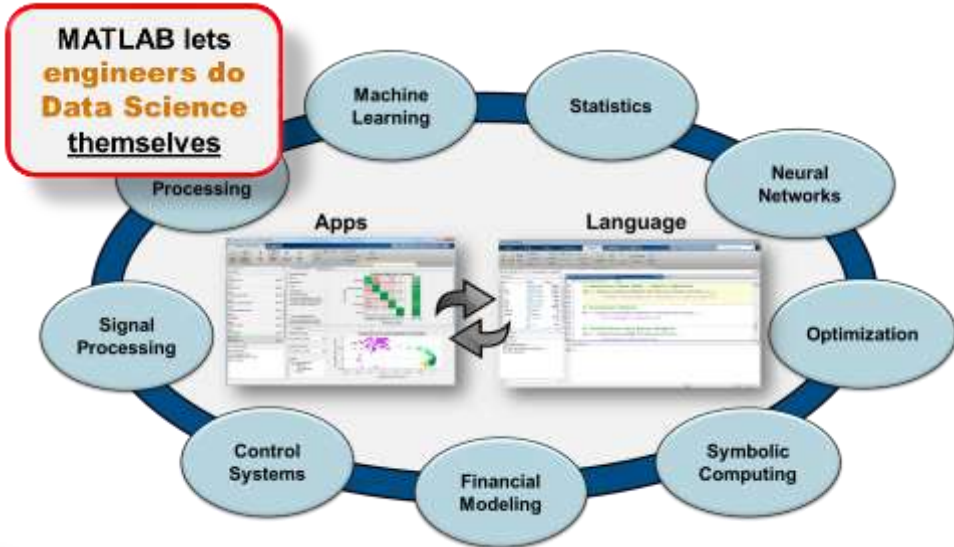
Develop Predictive Models

Integrate Analytics with Systems

MATLAB Analytics work with business and engineering data

MATLAB lets engineers do Data Science themselves

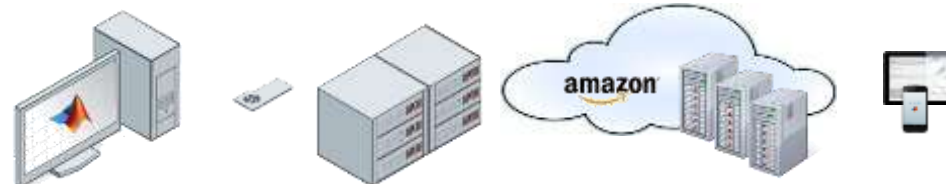
MATLAB Analytics run anywhere



- Microcontrollers
- DSP chips
- FPGAs
- Industrial automation systems

MATLAB Analytics Scale

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



- 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)

Additional Resources

Documentation:

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

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

The screenshot shows the documentation page for the Statistics and Machine Learning Toolbox. The header includes 'Documentation' and a search bar for 'R2016b Documentation'. A sidebar on the left lists 'CONTENTS' with sections for 'Getting Started', 'Descriptive Statistics and Visualization', 'Probability Distributions', and 'Hypothesis Tests'. The main content area features the title 'Statistics and Machine Learning Toolbox' with the version 'R2016b' and a brief description: 'Analyze and model data using statistics and machine learning'. It provides an overview of the toolbox's capabilities, such as descriptive statistics, Monte Carlo simulations, and machine learning algorithms. A 'Download a Free Trial' button is visible in the background.

Documentation Search R2016b Documentation

CONTENTS

Statistics and Machine Learning Toolbox R2016b

Analyze and model data using statistics and machine learning

Statistics and Machine Learning Toolbox™ provides functions and apps to describe, analyze, and model data. You can use descriptive statistics and plots for exploratory data analysis, fit probability distributions to data, generate random numbers for Monte Carlo simulations, and perform hypothesis tests. Regression and classification algorithms let you draw inferences from data and build predictive models.

For multidimensional data analysis, Statistics and Machine Learning Toolbox provides feature selection, stepwise regression, principal component analysis (PCA), regularization, and other dimensionality reduction methods that let you identify variables or features that impact your model.

The toolbox provides supervised and unsupervised machine learning algorithms, including support vector machines (SVMs), boosted and bagged decision trees, *k*-nearest neighbor, *k*-means, *k*-medoids, hierarchical clustering, Gaussian mixture models, and hidden Markov models. Many of the statistics and machine learning algorithms can be used for computations on data sets that are too big to be stored in memory.

Getting Started
Learn the basics of Statistics and Machine Learning Toolbox

Descriptive Statistics and Visualization
Data import and export, descriptive statistics, visualization

Probability Distributions
Data frequency models, random sample generation, parameter estimation

Hypothesis Tests
t-test, F-test, chi-square goodness-of-fit test, and more

Examples
[Functions and Other Reference](#)
[Release Notes](#)
[PDF Documentation](#)

The screenshot shows the 'Machine Learning with MATLAB' page. The header includes 'Machine Learning with MATLAB' and a search bar for 'Search MathWorks.com'. The main content area features a large image with a neural network visualization and a 'Download a Free Trial of Statistics and Machine Learning Toolbox' button. Below the image, there is a paragraph describing the use of machine learning in engineering and data science, followed by a list of capabilities: classification, regression, and clustering. To the right, there are two sidebars: one for 'Choosing the Best Classification Model and Avoiding Overfitting' with a 'Download white paper' link, and another for 'Explore Products for Machine Learning' listing various toolboxes like 'Statistics and Machine Learning Toolbox™', 'Neural Network Toolbox™', 'Computer Vision System Toolbox™', and 'Fuzzy Logic Toolbox™'.

Machine Learning with MATLAB Search MathWorks.com

Download a Free Trial of Statistics and Machine Learning Toolbox

Download trial

Engineers and data scientists work with large amounts of data in a variety of formats such as sensor, image, video, telemetry, databases, and more. They use machine learning to find patterns in data and to build models that predict future outcomes based on historical data. With MATLAB®, you have immediate access to prebuilt functions, extensive toolboxes, and specialized apps for [classification](#), [regression](#), and [clustering](#). You can:

- Compare approaches such as logistic regression, classification trees, support vector machines, ensemble methods, and **deep learning**.
- Use model refinement and reduction techniques to create an accurate model that best captures the predictive power of your data.

Choosing the Best Classification Model and Avoiding Overfitting
» [Download white paper](#)

Explore Products for Machine Learning

- [Statistics and Machine Learning Toolbox™](#)
- [Neural Network Toolbox™](#)
- [Computer Vision System Toolbox™](#)
- [Fuzzy Logic Toolbox™](#)

