Introduction to Machine Learning

Part I

Mehran Behjati

UNIVERSITI
KEBANGSAAN
MALAYSIA
National University of Malaysia

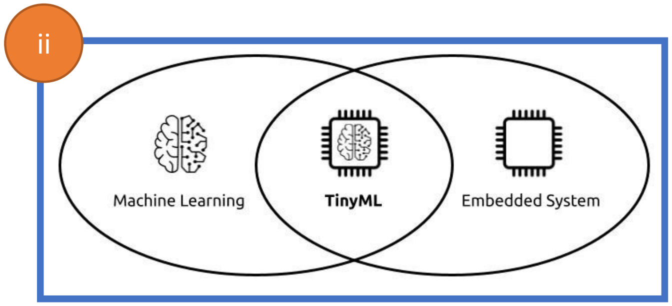
Computer
evolution
accuracy
Calculus
relationships
simplified
theoretically
humans
overlap
methods
Convolutional
labelled
technologies
training
neurons
patterns
approximates
Structured
understand
extracting
Receptive
ground-up
research
layers
complicated
intervention
technology
programs
scales
hierarchically
successful
Learning
definitions
pooling
Propositional
systems
explanation
concepts
Networks
Applications
experience
capabilities
Vision
building
field
CNNs
idea
Reinforcement
Feature
modern
algorithms
representation
influenced
Deep Learning
Intelligence
features
comprehension
machinery
deepening
problems
Higher-level
consume
models
input
programming
complex
model
available
hierarchical
identified
relates
Deep
Plateau
defined
hidden
differentiates
characteristics
gradually
levels
discovering
associated
evolve
confusion
processing
generalized
definition
Machine
layer
connected
relate
learner
inputs
deepness
traditional
knowledge
exposed
process
abstraction
set
filters
intelligent
classification
Unsupervised
Parallel
limitations
Machine Learning
power
automatically
computing
generalize
Predicate
brain
learn
unlabeled
identify
level
Neural
visual
hierarchy
Performance
evaluating
natural
Artificial Intelligence
advancing
lower-level
capability
understanding
identifying
deeper
Artificial
Data
implementing
artificially
Supervised
actions
techniques
architecture
Analysis
information
mathematical
considered
mathematicians
overlapping
comprehend

Layout

i

What is AI, ML, & DL

ii

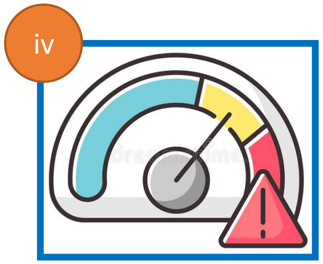


ML vs TinyML

iii

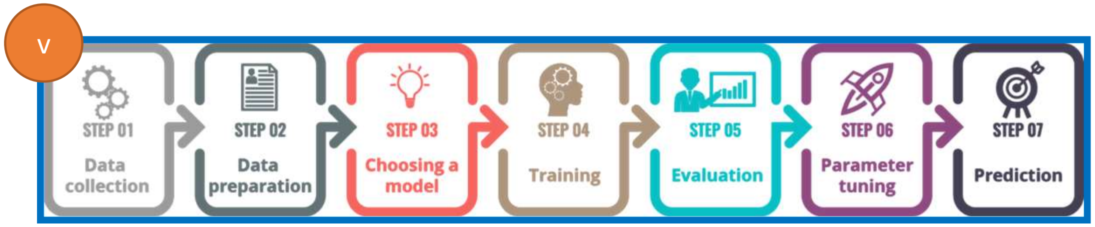
ML enablers

iv



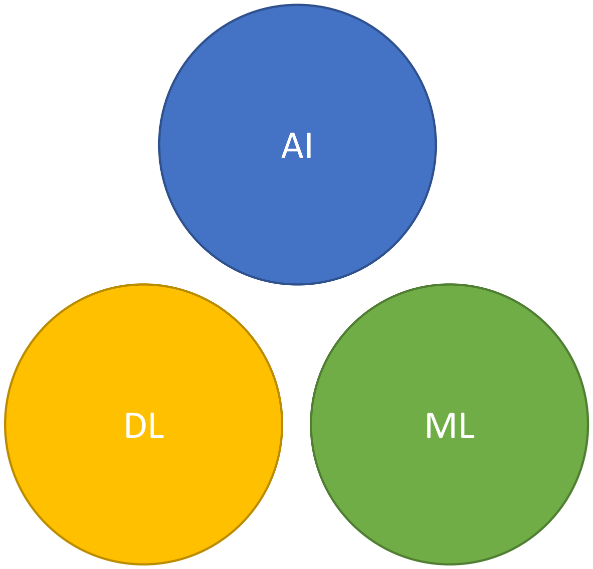
ML Limitations

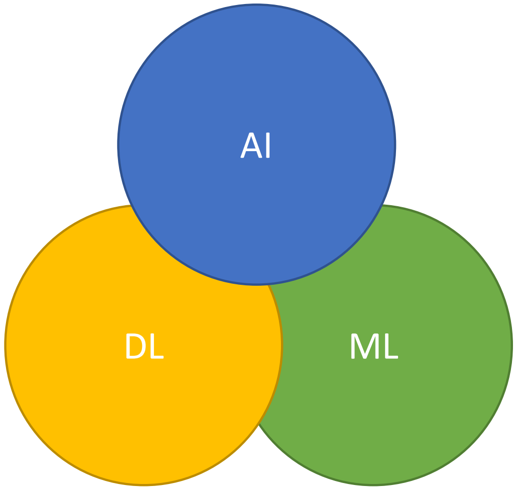
v

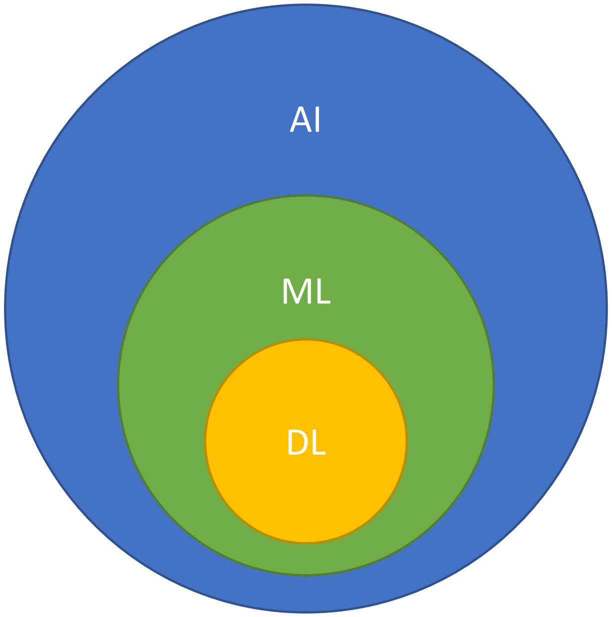


ML Process

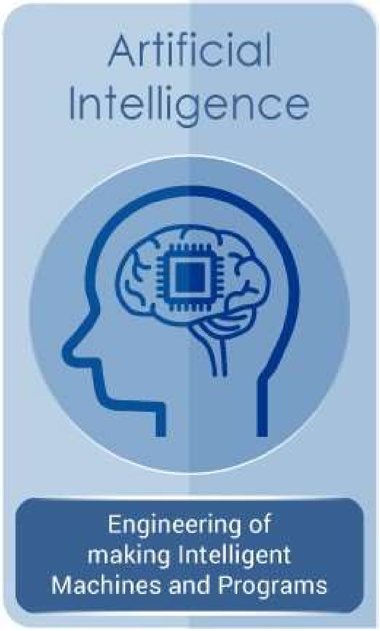
AI ML DL





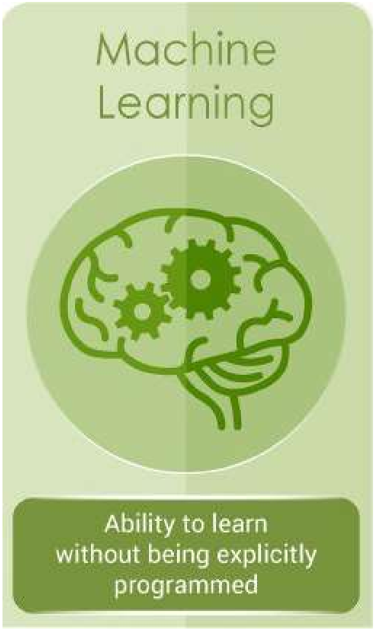


What is AI, ML & DL?



Artificial Intelligence

Engineering of making Intelligent Machines and Programs



Machine Learning

Ability to learn without being explicitly programmed

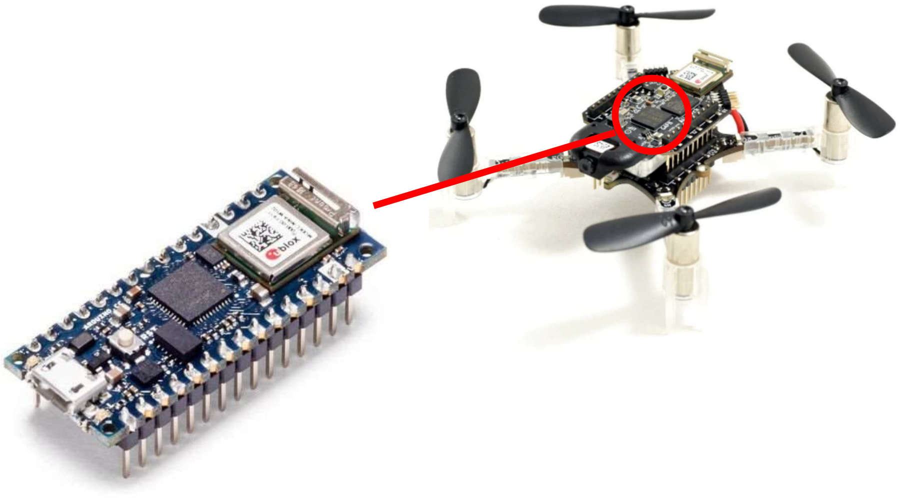
Deep Learning

Learning based on Deep Neural Network

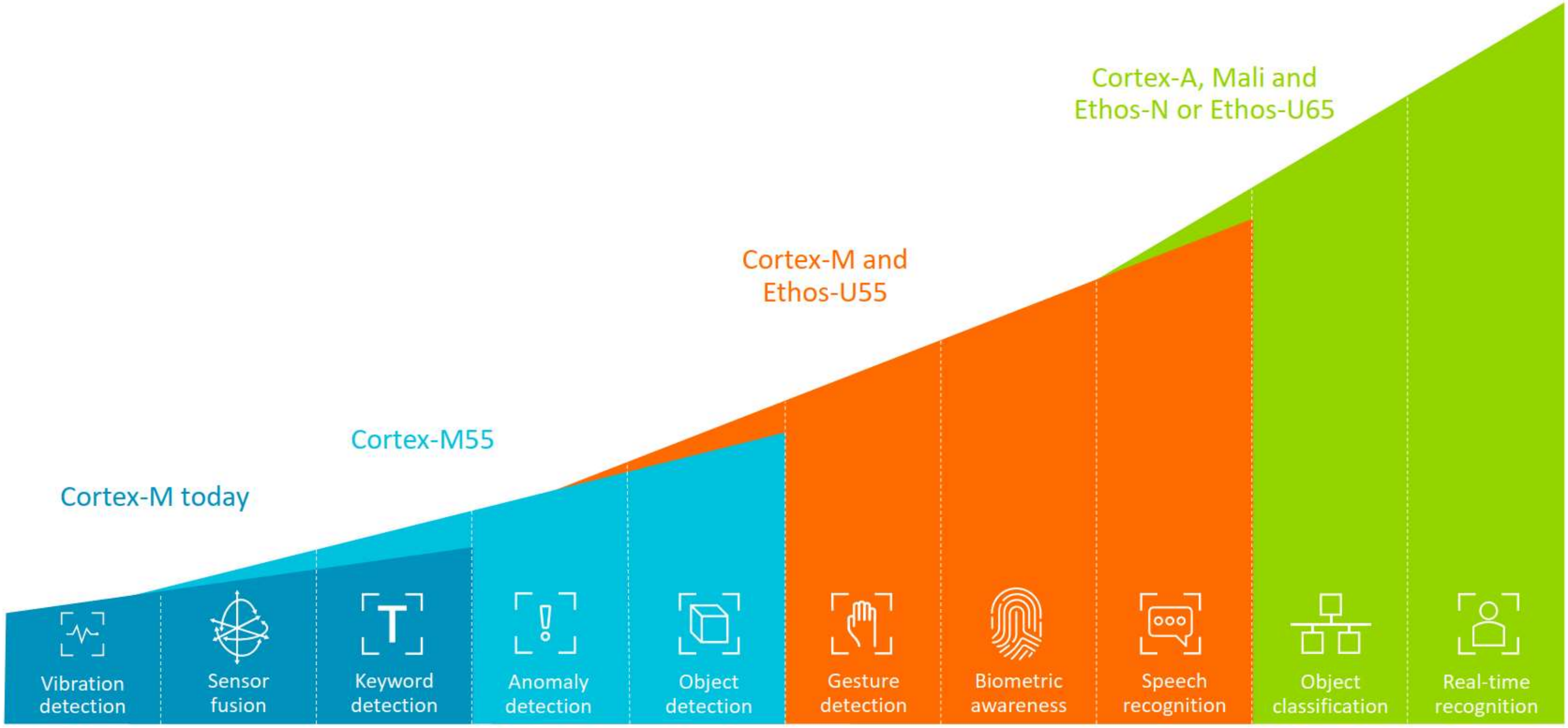
ML vs TinyML

- **ML**
  - More powerful
  - Longer processing
  - More power consumption

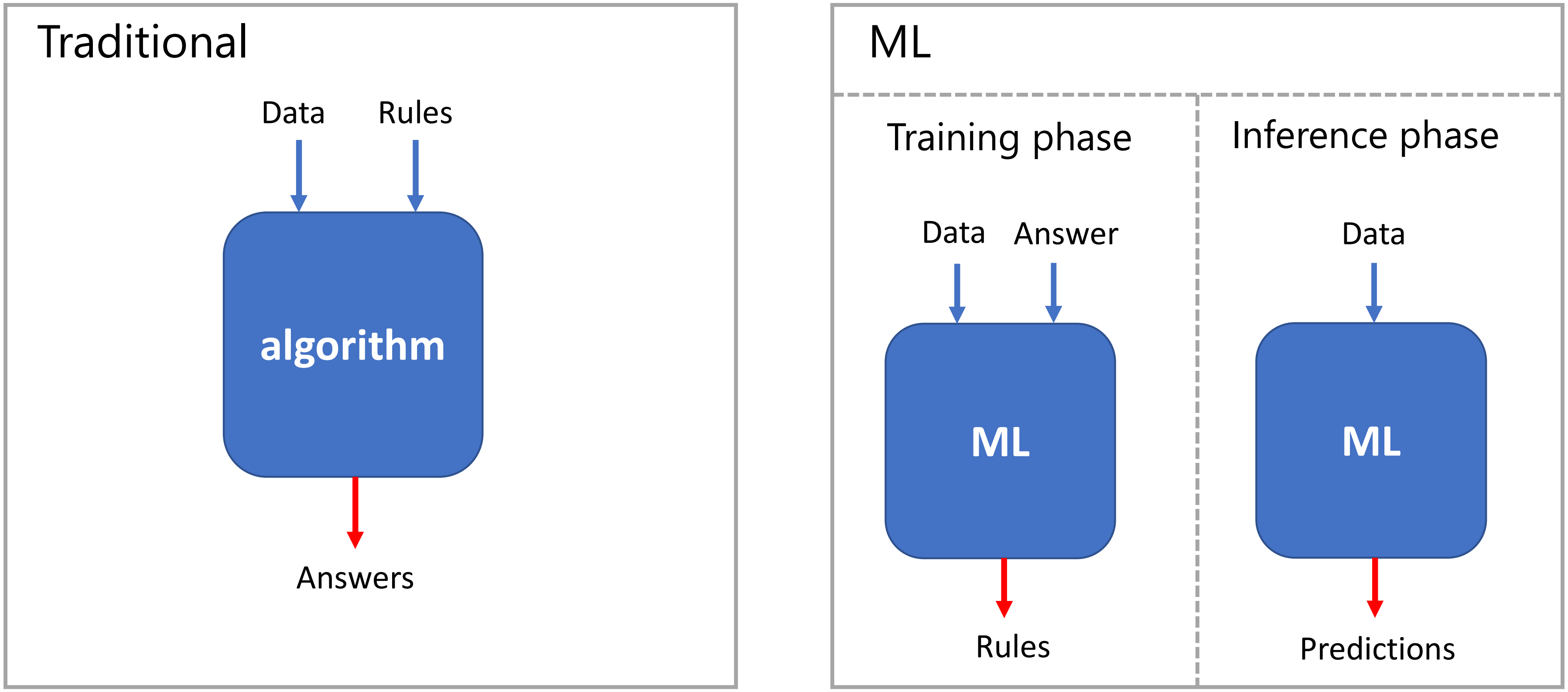
- **TinyML**
  - Less powerful
  - Light processing
  - Power efficient



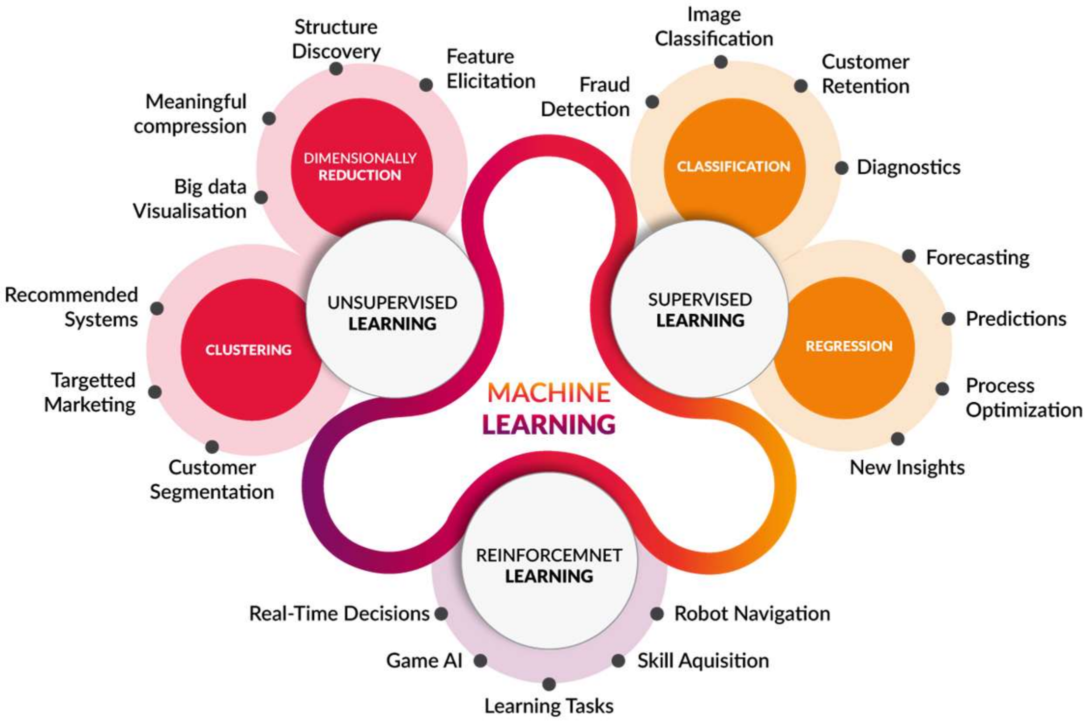
Some TinyML Applications



Traditional algorithms vs ML algorithms

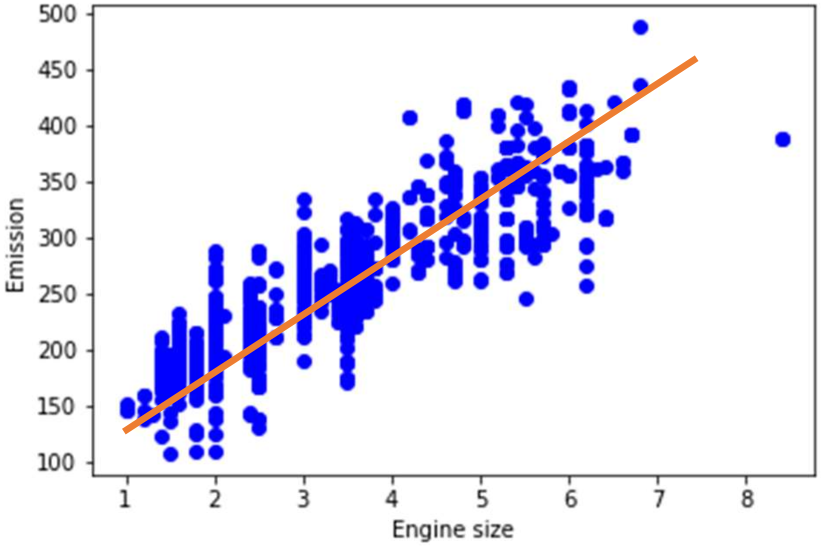


ML Categories

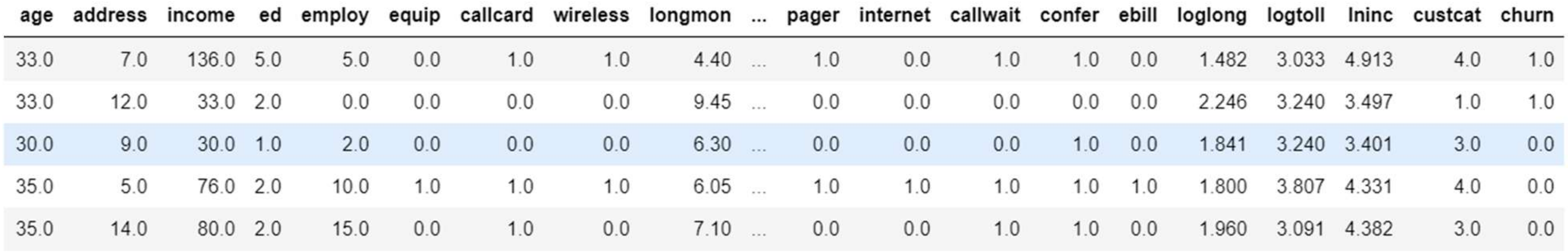


Regression

CO2


500
450
400
350
300
250
200
150
100
Emission
1
2
3
4
5
6
7
8
Engine size

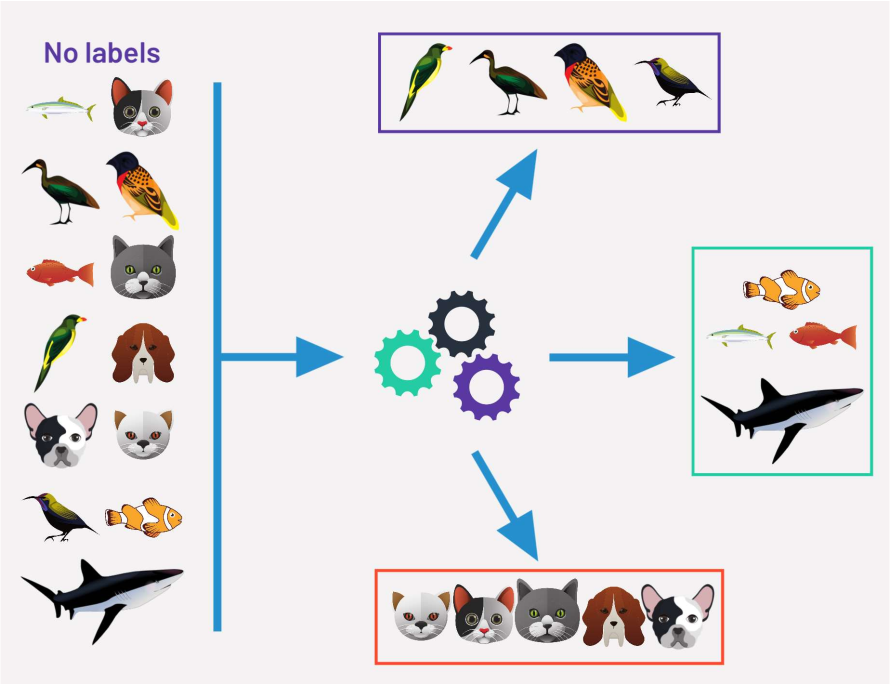
Classification



age	address	income	ed	employ	equip	callcard	wireless	longmon	...	pager	internet	callwait	confer	ebill	loglong	logtoll	lninc	custcat	churn
33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	4.40	...	1.0	0.0	1.0	1.0	0.0	1.482	3.033	4.913	4.0	1.0
33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	9.45	...	0.0	0.0	0.0	0.0	0.0	2.246	3.240	3.497	1.0	1.0
30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	6.30	...	0.0	0.0	0.0	1.0	0.0	1.841	3.240	3.401	3.0	0.0
35.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	6.05	...	1.0	1.0	1.0	1.0	1.0	1.800	3.807	4.331	4.0	0.0
35.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	7.10	...	0.0	0.0	1.0	1.0	0.0	1.960	3.091	4.382	3.0	0.0

Clustering

- It is unsupervised, so there is no label



ML Enablers

Data availability

Algorithm advancement,

Optimized software library & CMSIS-NN optimization algorithms

Open source framework, TensorFlow Lite (TinyML)

Computational power

TPU, GPU

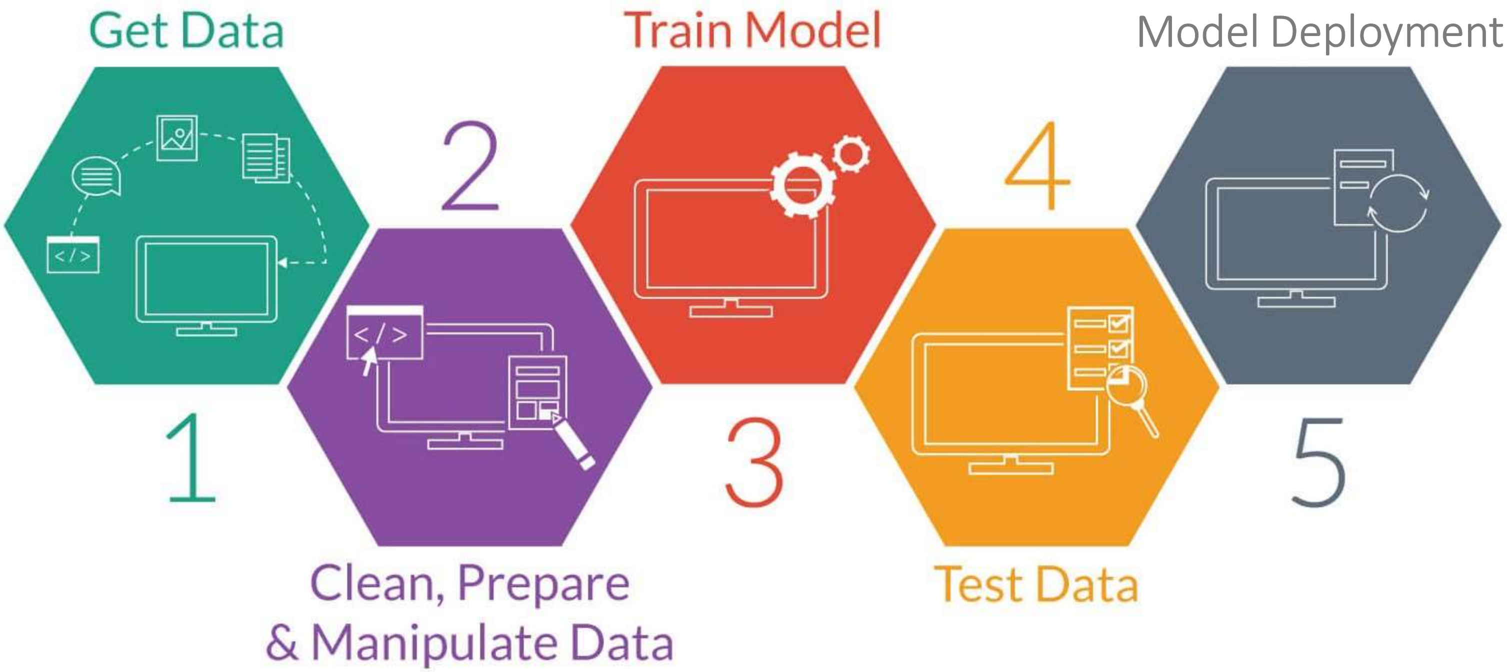
Cortex-M, microNPUs (TinyML)

Public/industry interest/demand

ML Limitations

- **Probabilistic vs Deterministic**
  - ML by its nature is probabilistic
- **Data**
  - Lack of data in some cases / lack of good data
- **Interpretability**
- **Misapplications**
- **Power and Cycle constraints (TinyML)**
  - need for HW & SW optimization
  - some already addressed e.g., energy efficient Arm Cortex-M55, TensorFlow Lite, SW optimization

ML Process


Get Data
1
2
Clean, Prepare
& Manipulate Data
Train Model
3
4
Test Data
Model Deployment
5

Data & ML

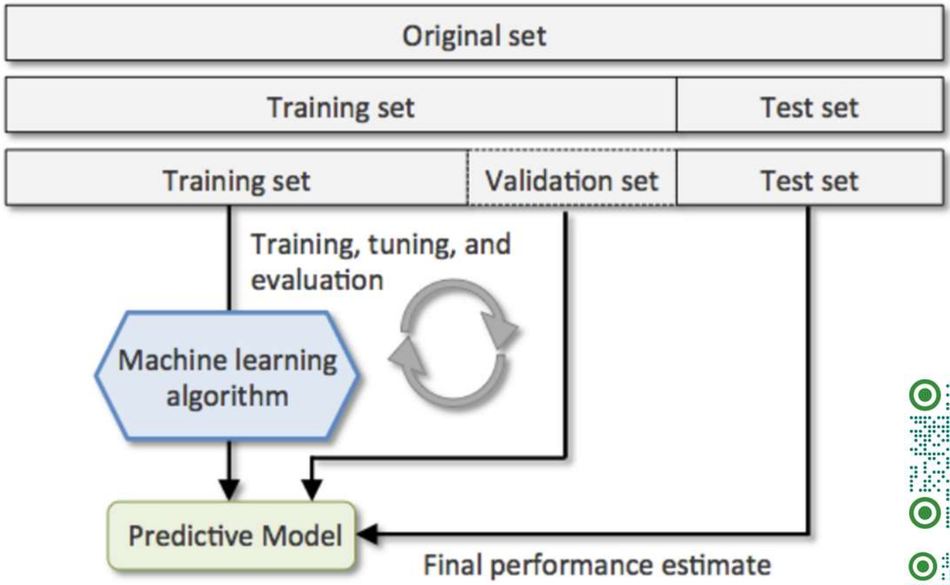
How to build a dataset for ML

Why do we need data?

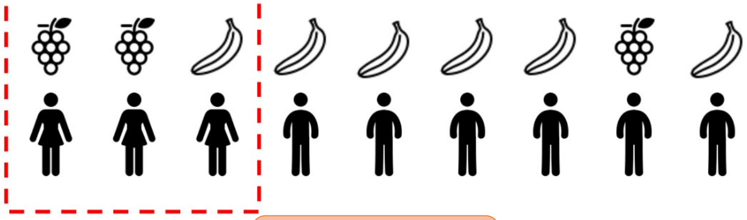
Training dataset: The sample of data used to fit the model.

Validation Dataset: The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters.

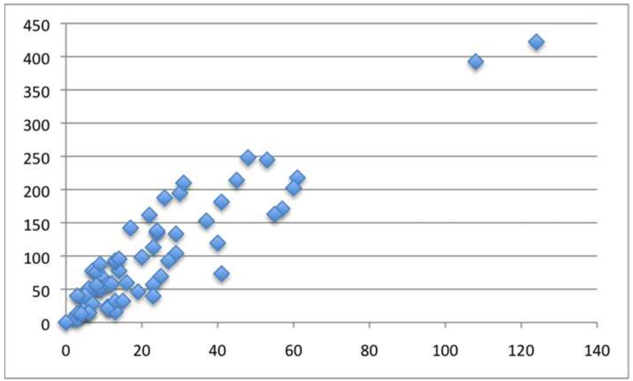
Test Dataset: The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.


Original set
Training set
Test set
Training set
Validation set
Test set
Training, tuning, and evaluation
Machine learning algorithm
Predictive Model
Final performance estimate

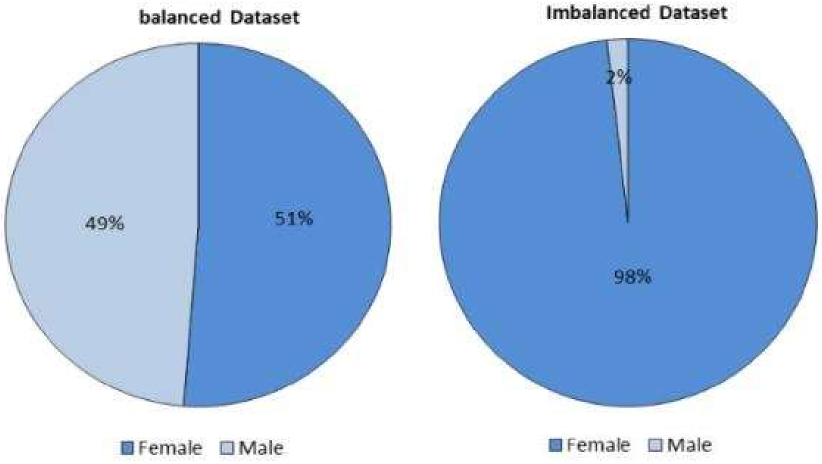
Some Data Collection Issues



Bias



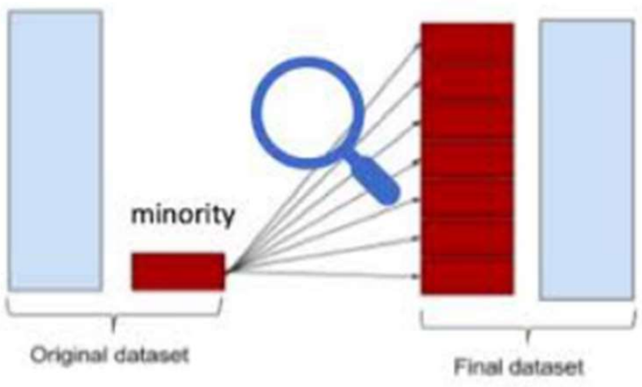
Small

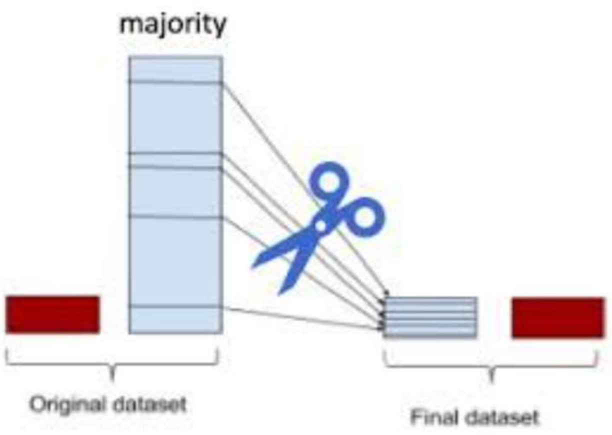


Unbalanced

Some solutions for unbalanced dataset

- Up/over sampling
- Down/under sampling
- Feature selection

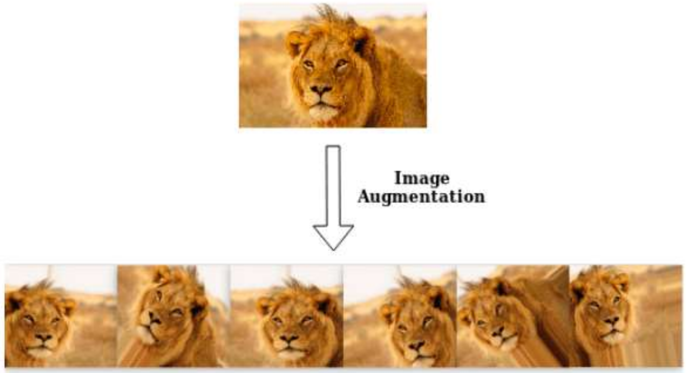




Some solutions for small size dataset

“what if we only had more data”

- Collect more data.
  - Is it always possible?
- Generate synthetic data (depends upon use case and the final goal)
  - Can we do it manually?
  - Can machine generate useful information?



Some solutions for bias dataset

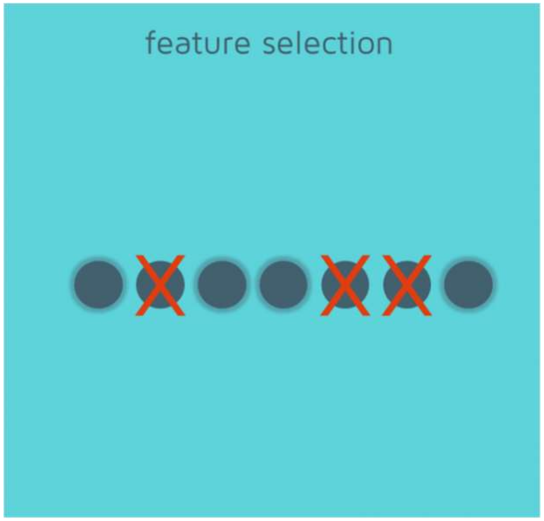
- Sample bias
  - Covering all the cases you expect your model to be exposed to.
- Exclusion bias
  - Investigate before discarding feature
  - Ask a domain expert
- Observer and prejudice bias
  - Ensure observers are well trained
  - Having clear rules
- Measurement bias
  - Having multiple measuring devices

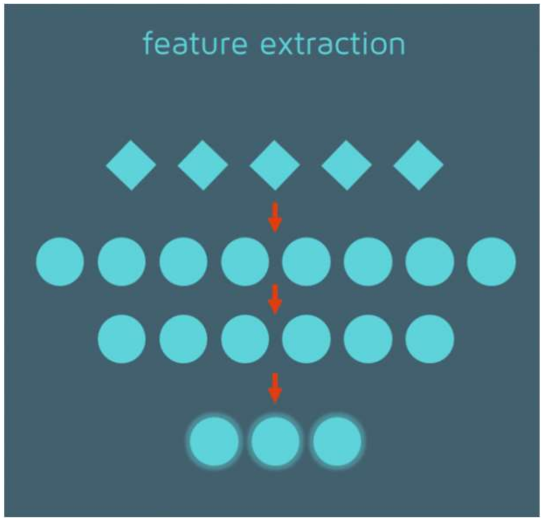
Data Cleansing & Preprocessing

- **Format:** The data might be spread in different files
- **Data Cleaning:** the goal is to deal with missing values and remove unwanted characters from the data
- **Remove unwanted data**
- **Manage unwanted outliers**
- **Handling missing data**

Feature Selection and Feature Extraction

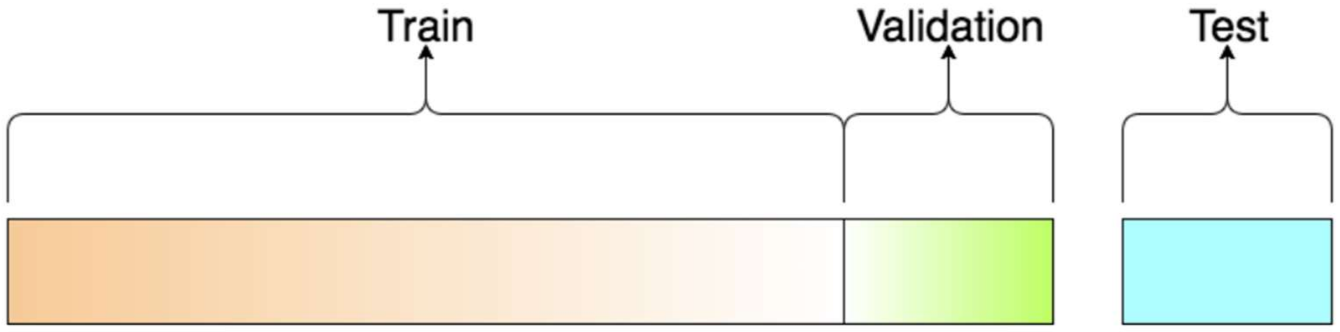
- **Feature:** Individual measurable property or characteristic of a phenomenon being observed
- **Selection:** Choosing a subset of the original pool of features.
- **Extraction:** Getting useful features from existing data.





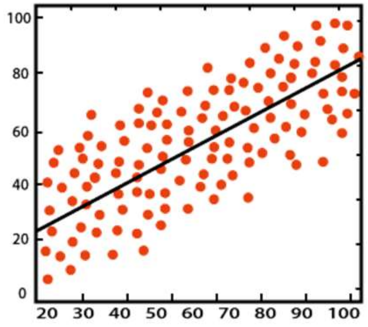
Train, Validation, and Test Sets

- **How to split?**
- **Mainly depends on:**
  - the total number of samples in your data
  - the actual model you are training

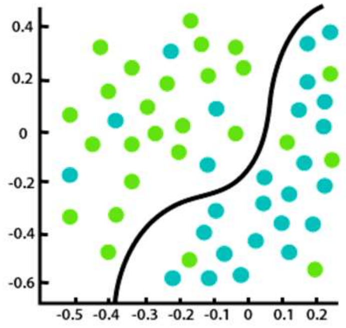


Training

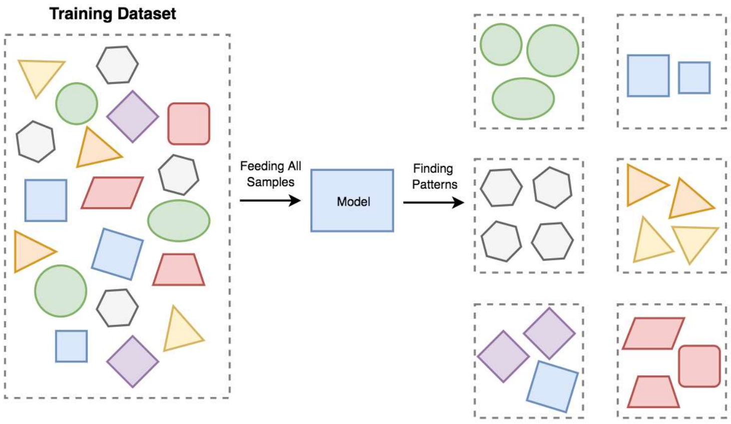
- **Regression models**
  - Linear regression
  - Polynomial
  - …
- **Classification models**
  - KNN
  - Decision tree
  - Logistic regression
  - SVM
  - …
- **Clustering models**
  - K-means
  - Hierarchy
  - DBSCAN
  - …



Regression



Classification



Training Dataset

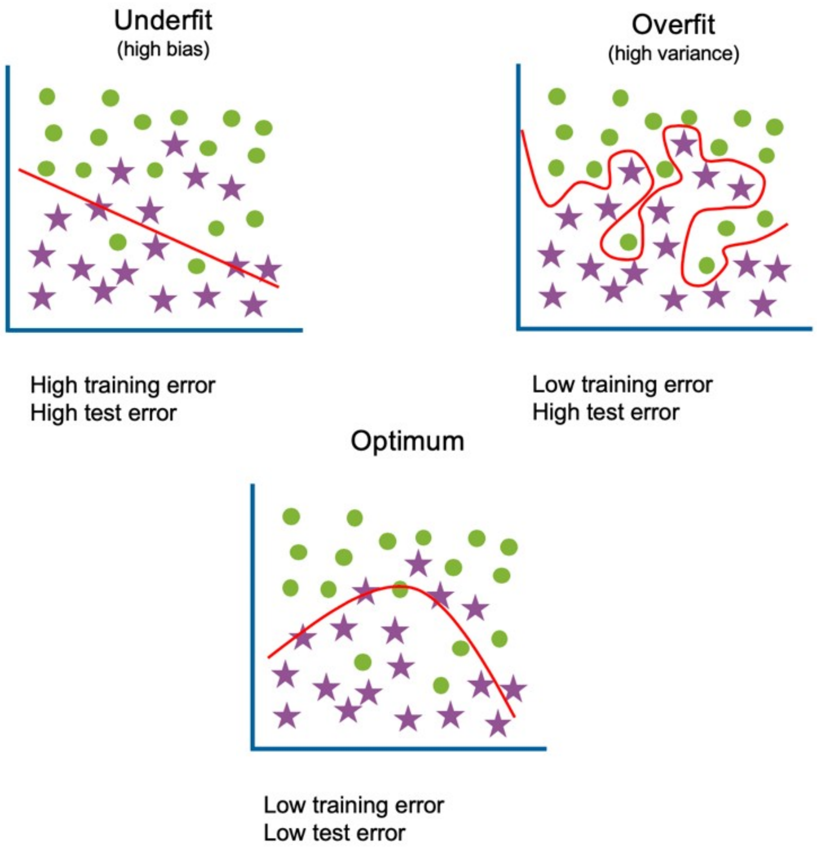
Feeding All Samples

Model

Finding Patterns

Overfitting & Underfitting

- **Underfitting:** Model performs poor under both training and test sets.
- Solutions:
  - Add more data
  - Try different features or more features
  - Try for longer
  - Reduce dropout nodes
  - Try a more complex model (increasing #layer & #neurons)
- **Overfitting:** Model predicts training data well but fails to generalize to test data.
- Solutions:
  - Add more data / data augmentation
  - Early stopping
  - Reduce model complexity (reducing #layer & #neurons)
  - Add dropout nodes

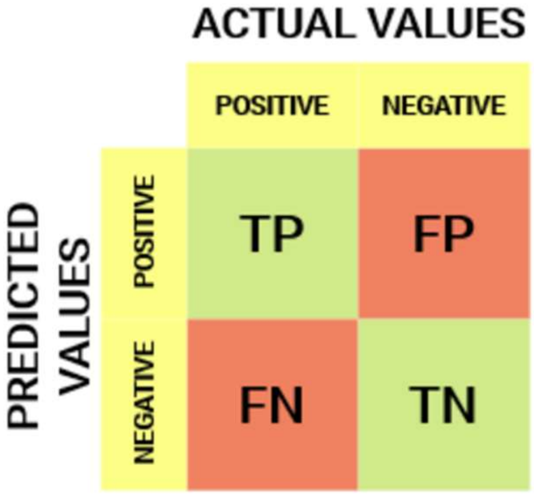


Evaluation

- **How we can evaluate the performance of our developed model?**
- **Metrices:**
  - Root Mean Squared Error (RMSE)
  - Mean Absolute Percentage Error (MAPE)
  - Median Absolute Error (MedAE)
  - Accuracy
  - Confusion matrix

Confusion Matrix

- A performance measurement for ML classification.
- Applicable for two or more class classification problems.
- Useful for measuring
  - $Recall = \frac{TP}{TP+F}$
  - $Precision = \frac{TP}{TP+FP}$
  - $F1 = 2 \times \frac{Precision \times Recall}{Precision \quad Recall}$



		ACTUAL VALUES	
		POSITIVE	NEGATIVE
PREDICTED VALUES	POSITIVE	TP	FP
	NEGATIVE	FN	TN

Resources

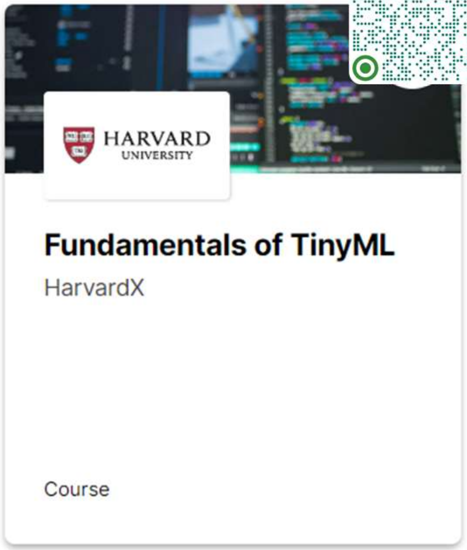
Edge Impulse

Introduction to Embedded Machine Learning

Skills you'll gain: Machine Learning

★ **4.8** (330 reviews)

Intermediate · Course · 1-4 Weeks

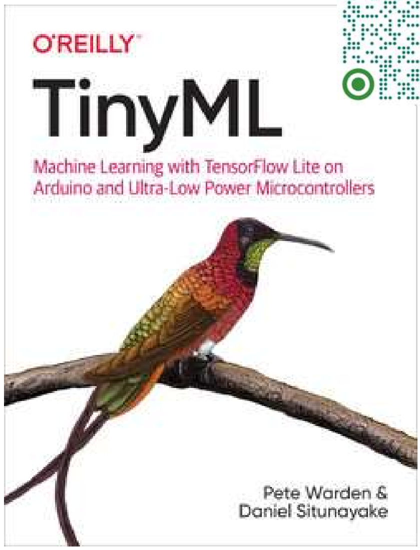


HARVARD UNIVERSITY

Fundamentals of TinyML

HarvardX

Course



O'REILLY

TinyML

Machine Learning with TensorFlow Lite on Arduino and Ultra-Low Power Microcontrollers

Pete Warden & Daniel Situnayake