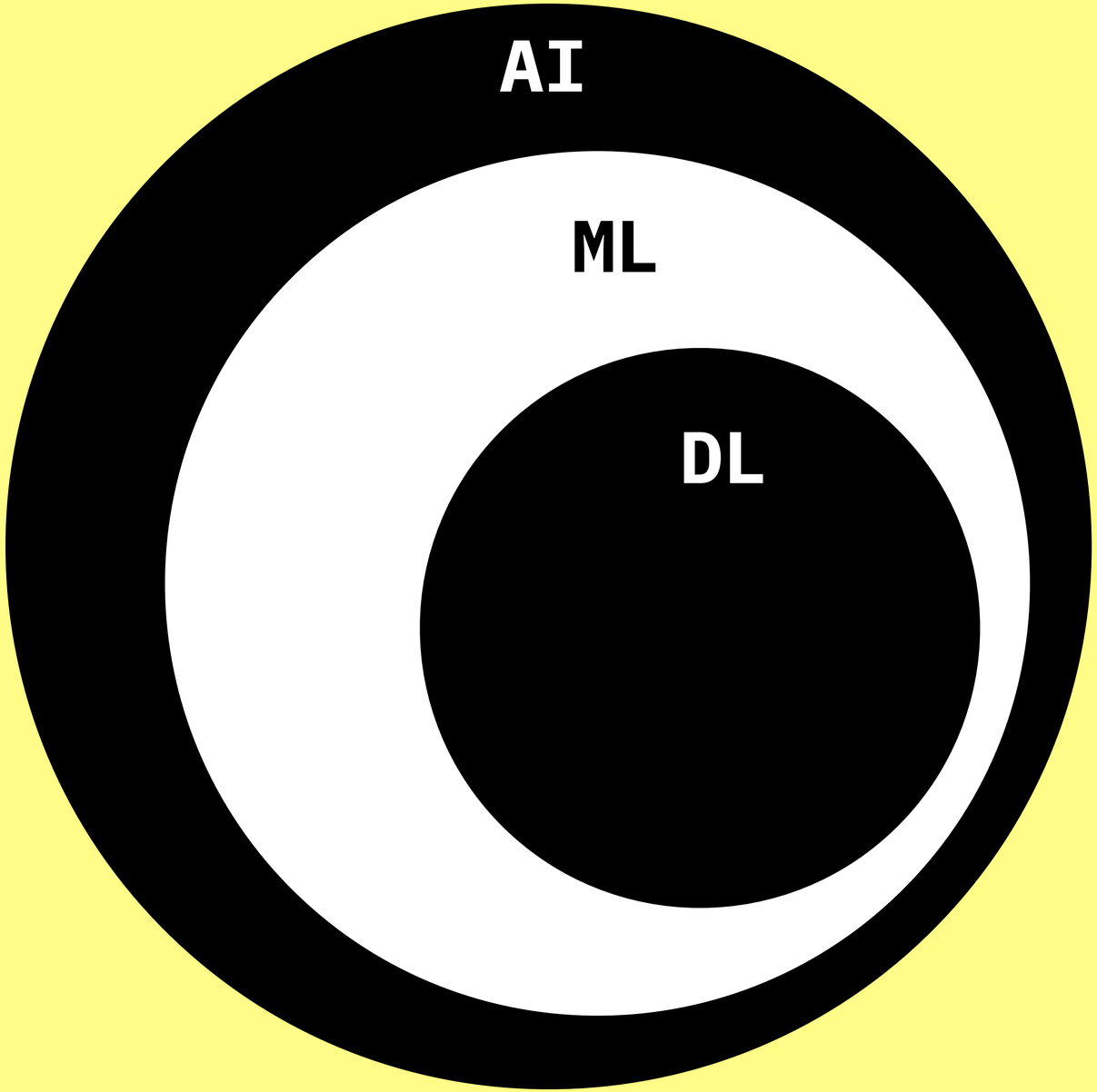
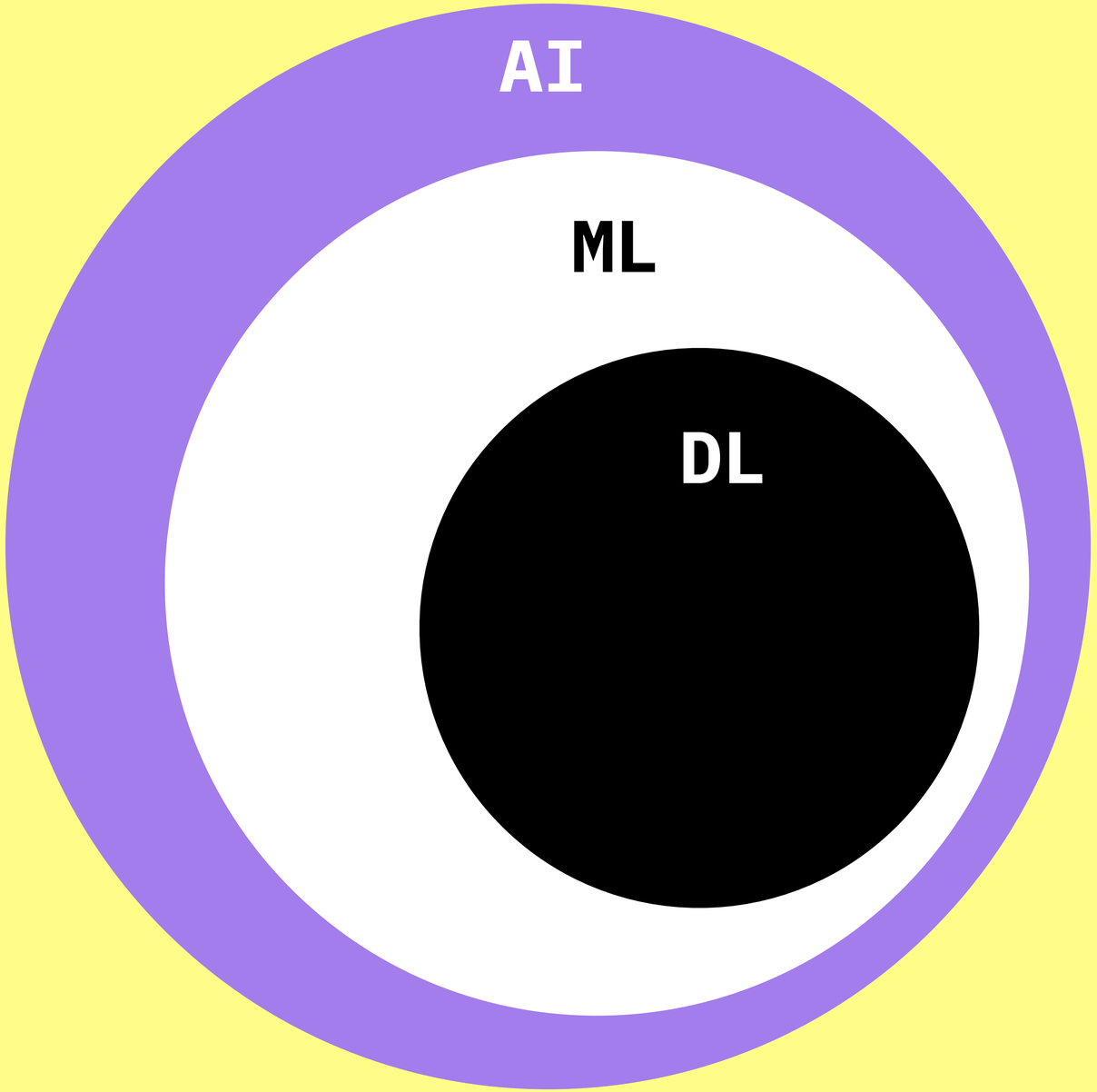
INTERACTION DESIGN ZHDK

MACHINE LEARNING

MUI HS21

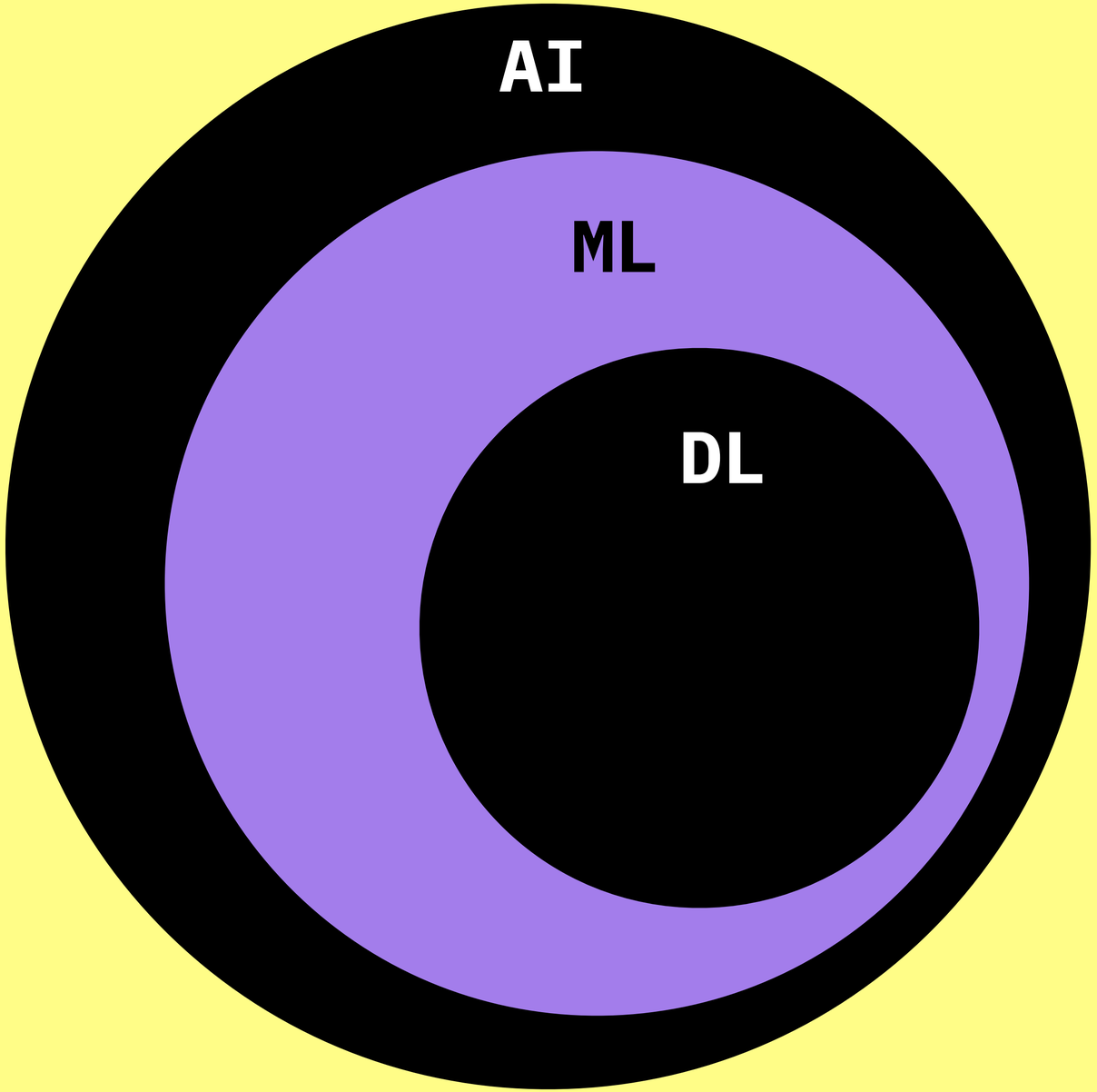

AI
ML
DL



ARTIFICIAL INTELLIGENCE*

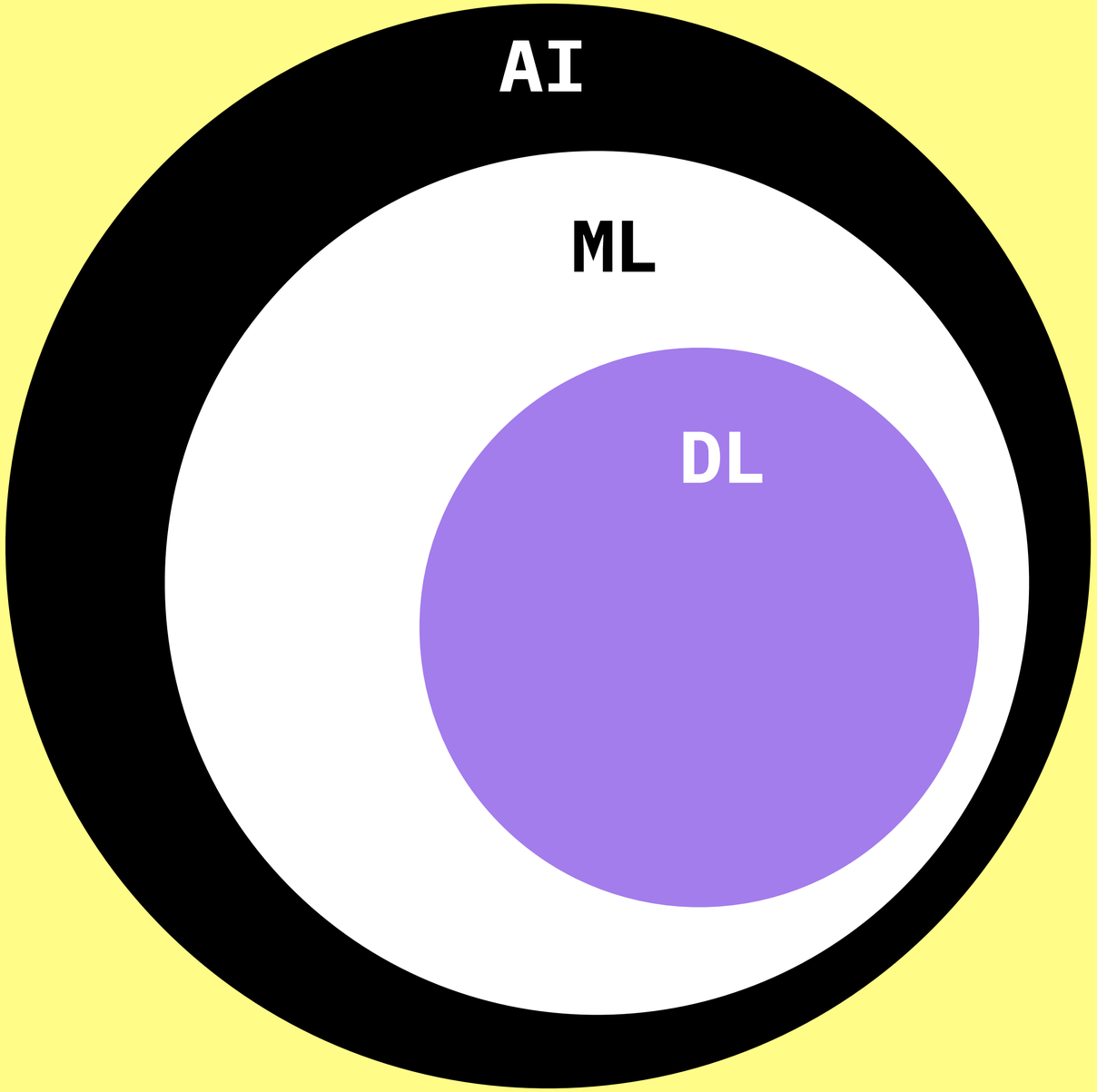
Technologies that are able to **perform specific tasks** as well as, **or better than**, we humans can.

*Narrow AI



MACHINE LEARNING

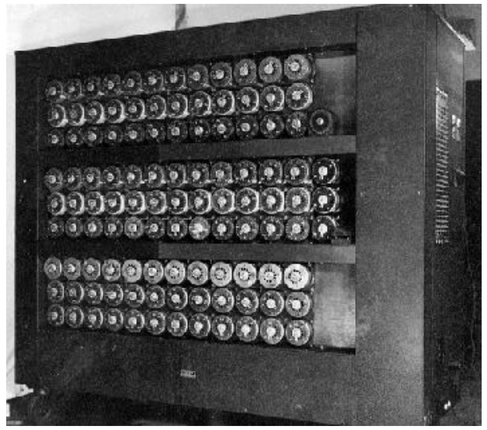
Uses **statistical techniques** to give computer systems the ability to learn (i.e., progressively improve performance on a specific task) with data, **without being explicitly programmed.**



DEEP LEARNING
Uses **multi-layered artificial neural networks** to deliver state-of-the-art accuracy.

Can automatically **learn features or representations from data** such as images, video or text.

TIMELINE

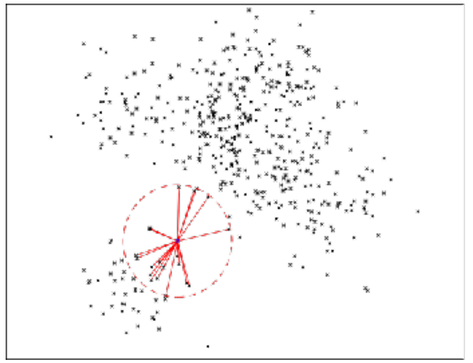


1950
Turing Test

1952
Checkers

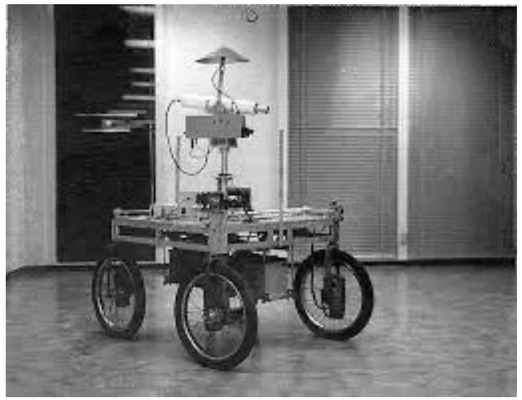
1957
The Perceptron

1967
The Nearest Neighbour



TIMELINE

1974-1980
AI WINTER



1979
The
Stanford
Cart

1996
IBM Deep Blue

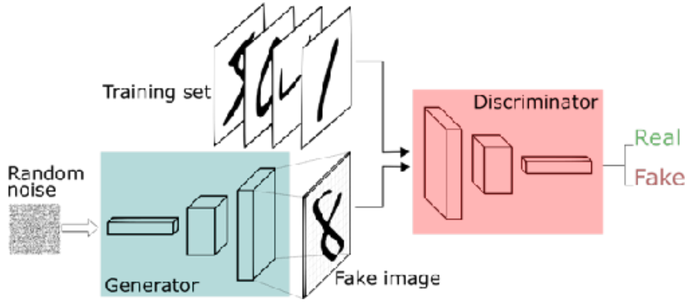
2002
Torch

TIMELINE

2006
Deep Learning

2011
IBM Watson

2012
ImageNet



2014
GANs

A Fast Learning Algorithm for Deep Belief Nets

Geoffrey E. Hinton
hinton@cs.toronto.edu
Simon Osindero
osindero@cs.toronto.edu
Department of Computer Science, University of Toronto, Toronto, Canada M5S 3G4

TIMELINE

2016
AlphaGo

2017
Deepfake

2018
Tensorflow.js
ml5.js

ml5

?

MACHINE LEARNING = BUZZWORD

Don't build a machine learning model where a simpler approach might succeed just as well!

TYPES

Supervised learning

- regression: predict numerical values
- classification: predict category

Unsupervised learning

- clustering: group data according to "distance"
- association: find frequent co-occurrences
- link prediction: discover relationships in data
- data reduction: many features to fewer features

Reinforcement learning

- Trial and error: achieve a goal in an uncertain, potentially complex environment

TYPES

Supervised learning

- regression: predict numerical values
- classification: predict category

The training data + the output we want to obtain (labels) are provided

Unsupervised learning

- clustering: group data according to "distance"
- association: find frequent co-occurrences
- link prediction: discover relationships in data
- data reduction: many features to fewer features

Data for training is provided, the desired output is not provided

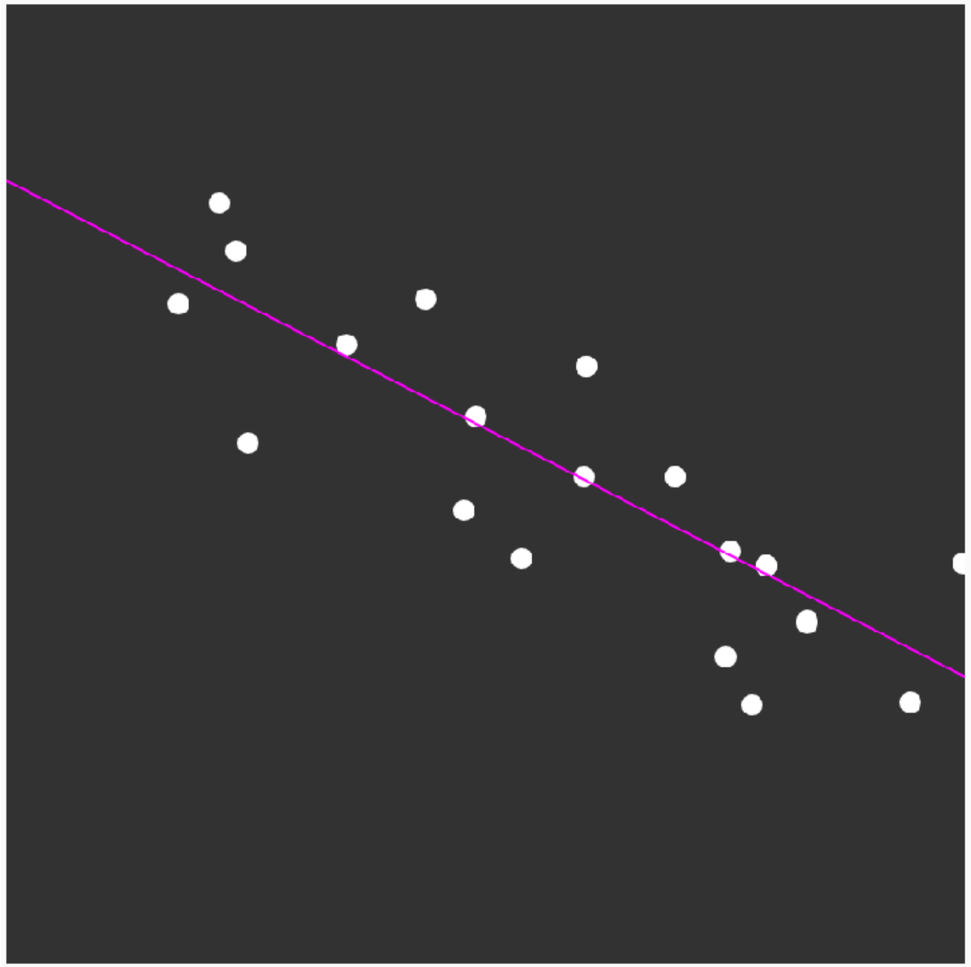
Reinforcement learning

- Trial and error: achieve a goal in an uncertain, potentially complex environment

Rewards, when certain tasks are performed correctly are provided

SUPERVISED

REGRESSION



https://editor.p5js.org/pzybinska/sketches/T5aBQ3tXs

SUPERVISED

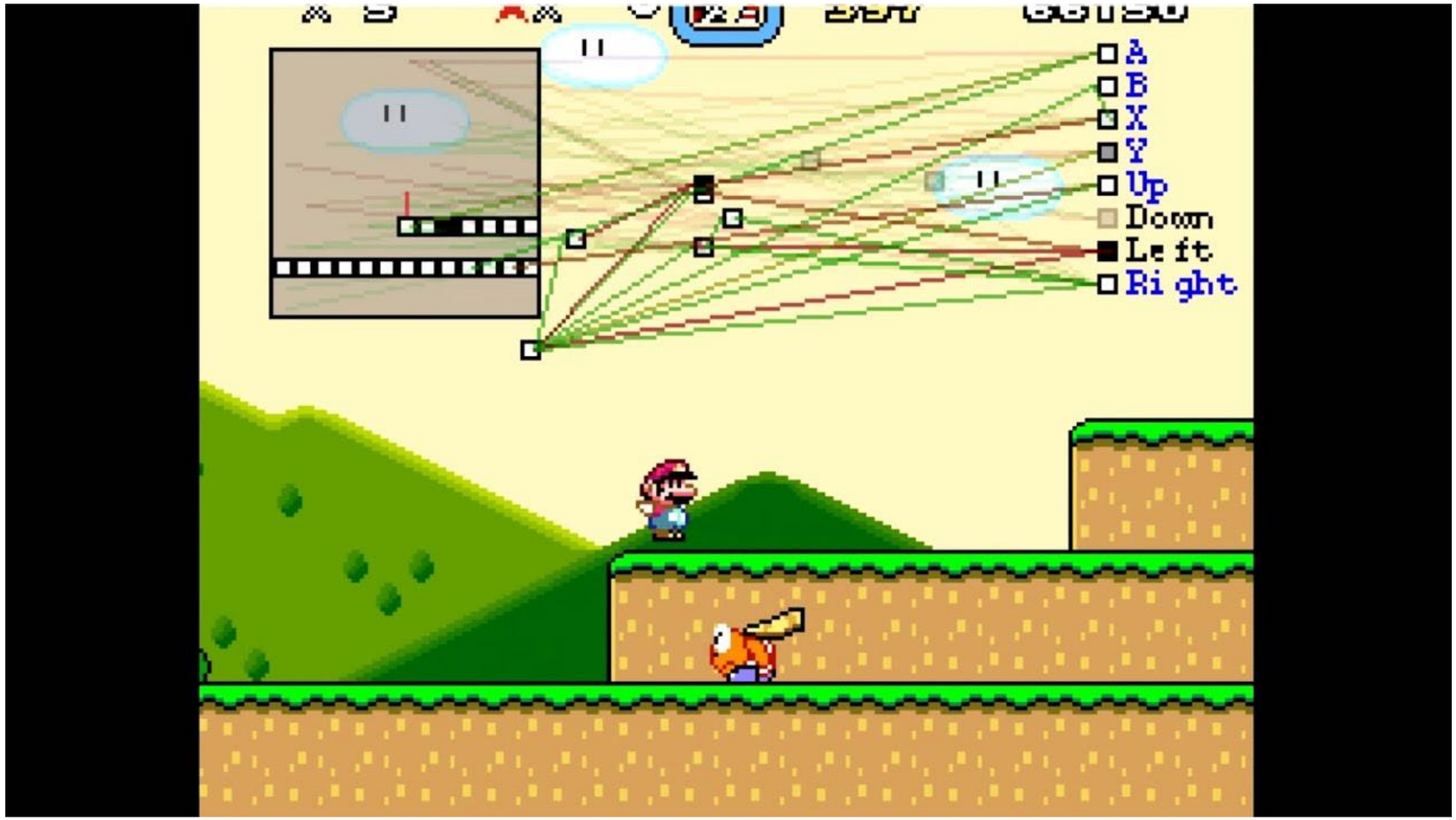
CLASSIFICATION

https://editor.p5js.org/AndreasRef/sketches/H1L-KrzFQ

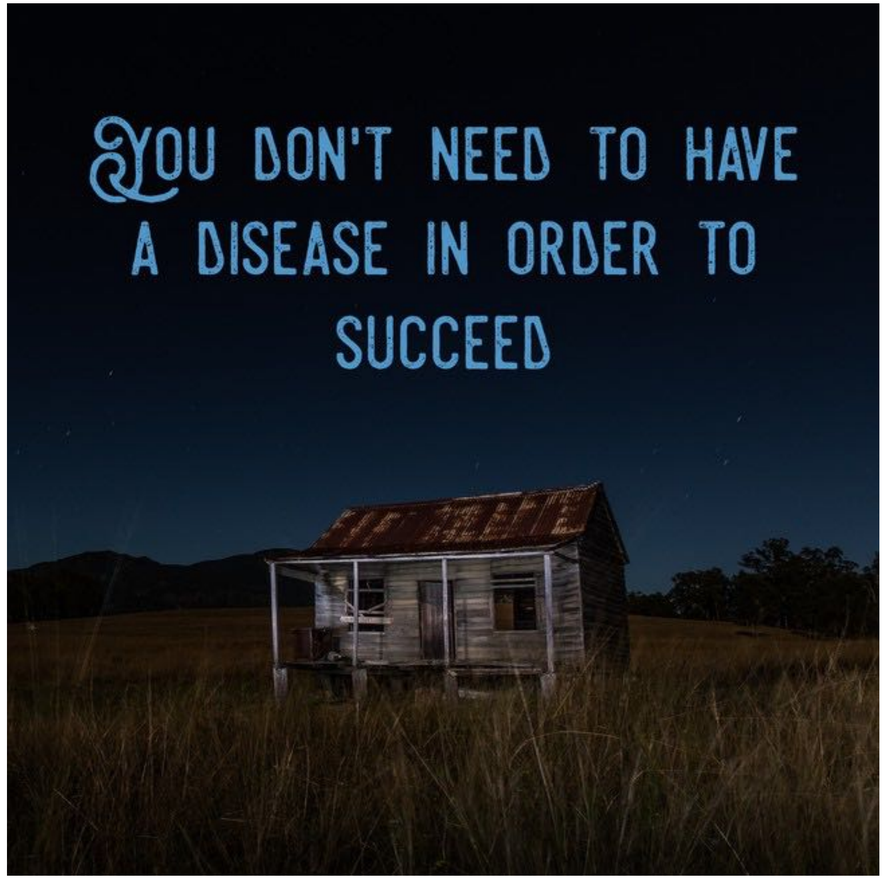
UNSUPERVISED

https://www.youtube.com/watch?v=yji0t6KS7Qo

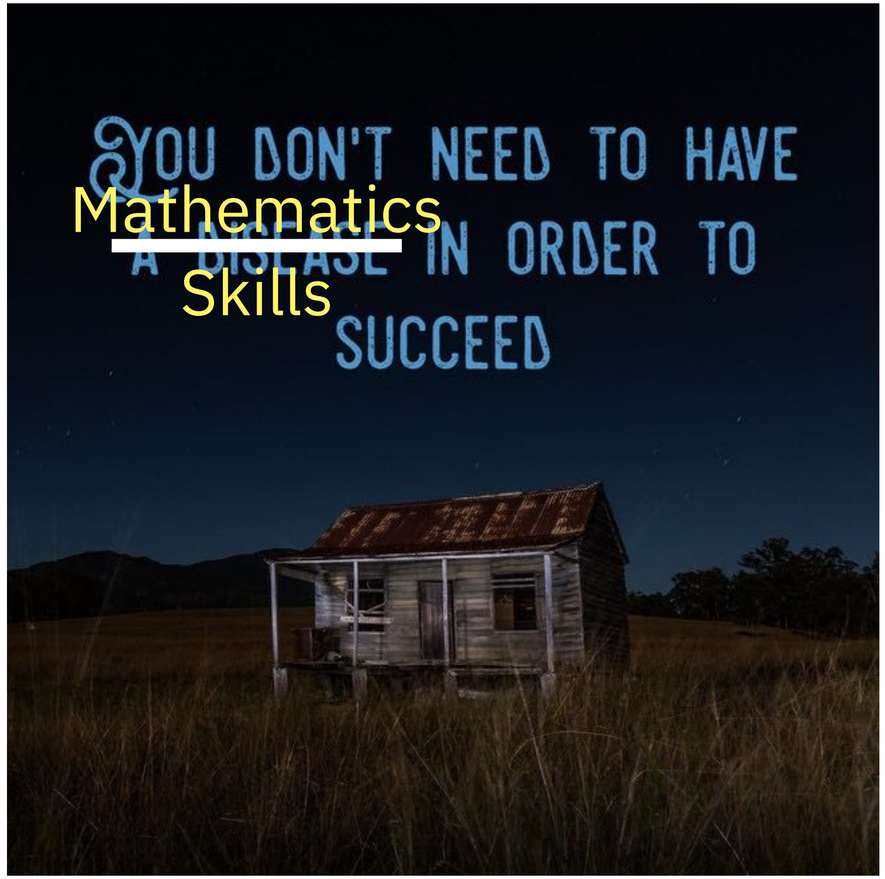
REINFORCMENT



https://www.youtube.com/watch?v=qv6UVOQ0F44



https://inspirobot.me/



~~Mathematics~~

Skills

https://inspirobot.me/

INTERACTION DESIGN ZHDK

ML5.JS

MUI HS21

P5.JS

```
<script src="p5.sound.js"></script>

function setup() {
  createCanvas(640, 480);
}

function draw() {
  let x = 80; //or var, const
  let y = 80;
  if (mouseIsPressed) {
    fill(0);
  } else {
    fill(255);
  }
  ellipse(mouseX, mouseY, x, y);
}
```

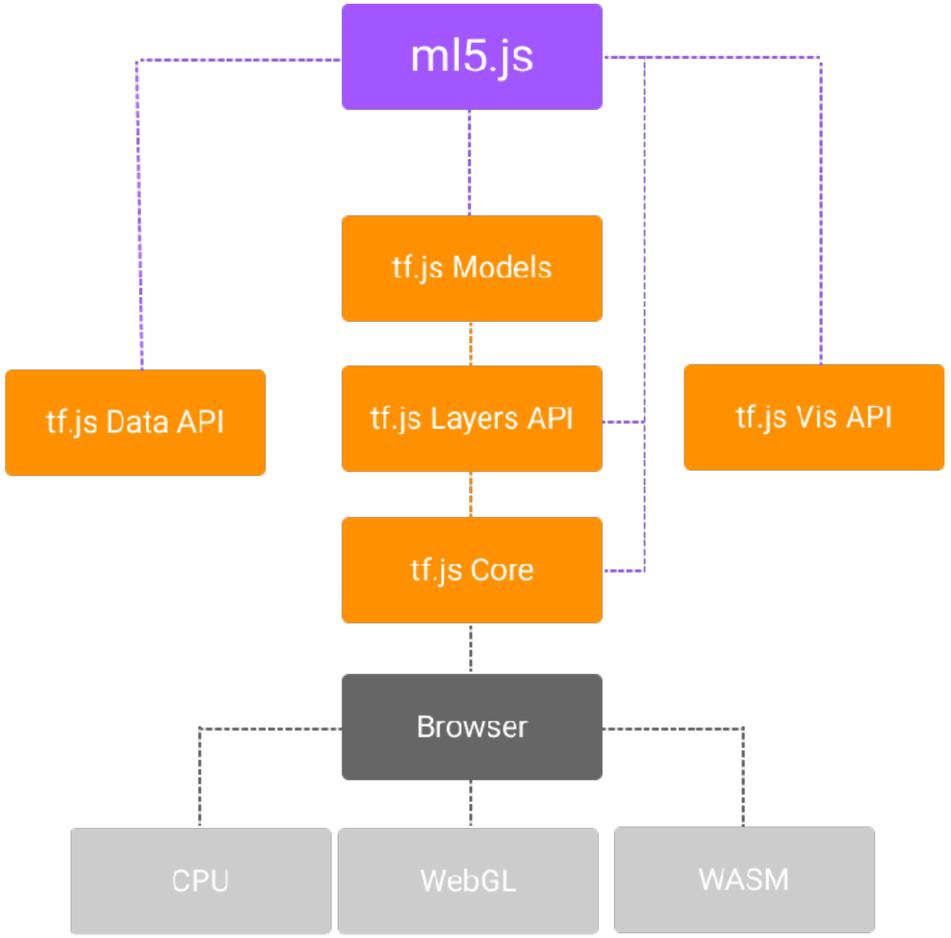
```
import processing.sound.*;

void setup() {
  size(640, 480);
}

void draw() {
  int x = 80;
  int y = 80;
  if (mousePressed) {
    fill(0);
  } else {
    fill(255);
  }
  ellipse(mouseX, mouseY, x, y);
}
```

ML5.JS

ml5.js provides access to machine learning algorithms and models in the browser, building on top of **TensorFlow.js**.



ML5.JS

Like any other javascript library it can be used for development of mobile apps with Angular.js, React.js, Vue.js, etc.

```
<script src="https://unpkg.com/ml5@latest/dist/ml5.min.js"></script>
```

Github Repository

STRUCTURE

Call ml5 function

```
const myClassifier = ml5.imageClassifier('MobileNet', modelReadyFunction);
```

Apply ml5 function

```
function modelReadyFunction(){
  myClassifier.classify(video, gotResultsFunction);
}
```

Do something with the result

```
function gotResultsFunction(error, results) {
  // an array of objects with "label" and "confidence"
  // [ { label: 'human', confidence: 0.74 } ]
  console.log(results);
}
```

MODELS

IMAGE

imageClassifier
ObjectDetector
poseNet
BodyPix
UNET
YOLO
StyleTransfer
Pix2Pix
CartoonGAN
FaceApi
Facemesh
Handpose
CVAE
DCGAN
SketchRNN

SOUND

Pitch Detection
SoundClassifier

TEXT

CharRNN
Word Vectorization
Sentiment
UniversalSentenceEncoder

HELPERS

NeuralNetwork
FeatureExtractor
KNNClassifier
KMeans

MODELS

imageClassifier('MobileNet')

```
const classifier = ml5.imageClassifier('MobileNet');

classifier.classify(video, gotResult);

function gotResult(error, result) {
  console.log(result);
}
```

My guess is a toaster.

My confidence is 0.12.

MODELS

PoseNet

```
const posenet = ml5.poseNet(video);

posenet.on('pose', function(results) {
  poses = results;
});

function draw() {
  if (poses.length > 0) {
    circle(poses[0].nose.x, poses[0].nose.y);
  }
}
```

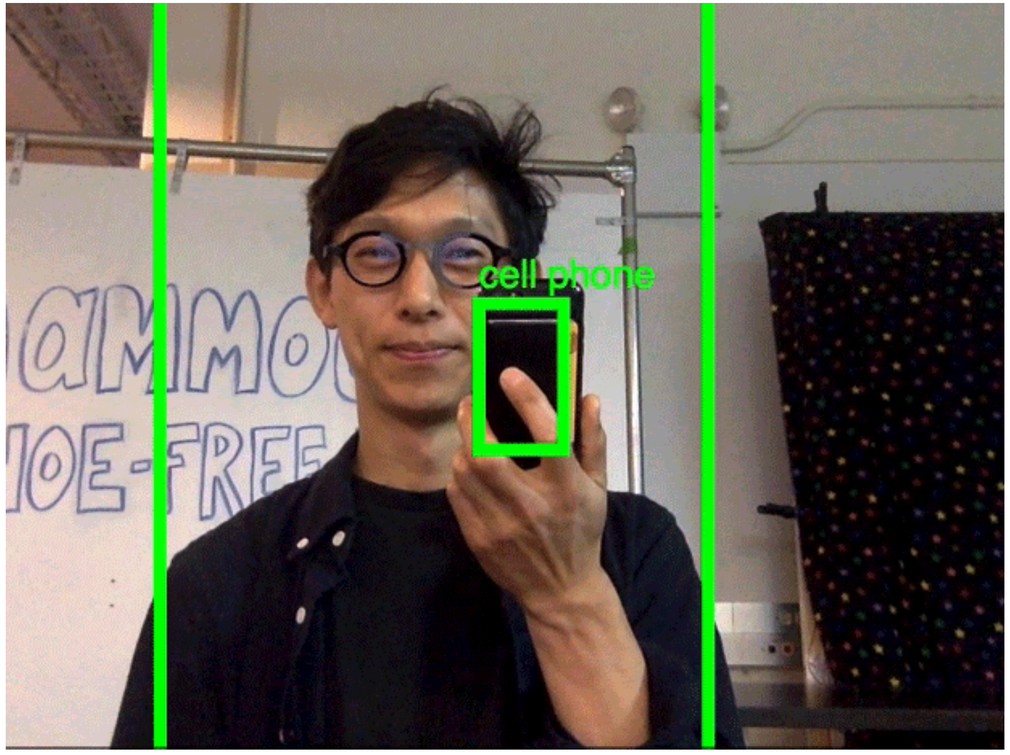
MODELS

ObjectDetector (YOLO or CocoSSd)

```
const classifier = ml5.imageClassifier('MobileNet');

classifier.classify(video, gotResult);

function gotResult(error, result) {
  console.log(result);
}
```



MODELS

SoundClassifier('SpeechCommands18w')

Label: zero
Confidence: 0.99

```
let classifier = ml5.soundClassifier('SpeechCommands18w');

classifier.classify(gotResult);

function gotResult(error, result) {
  labelDiv.html(result[0].label);
  confidenceDiv.html(result[0].confidence);
}
```

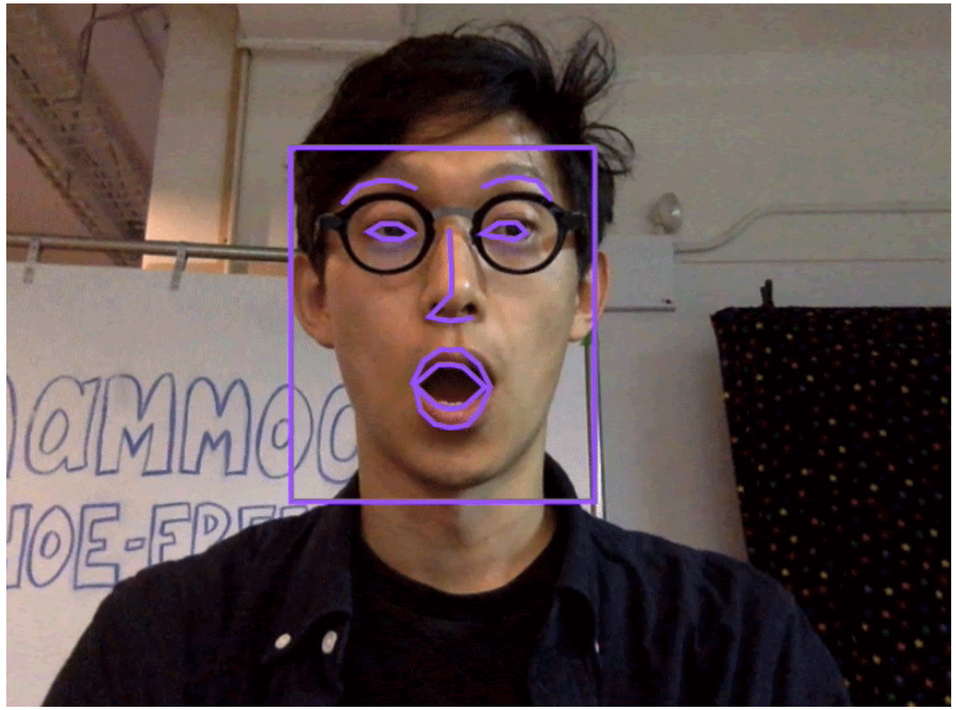
MODELS

Face-Api

```
const faceapi = ml5.faceApi();

faceapi.detect(video, gotResults);

function gotResults(error, results) {
  drawLandmarks(results);
}
```



MODELS

KNNClassifier + FeatureExtractor

```
const knnClassifier = ml5.KNNClassifier();

const featureExtractor =
ml5.featureExtractor('MobileNet', modelReady);

const features = featureExtractor.infer(myImg);

knnClassifier.addExample(features, label);

knnClassifier.classify(features, (err, result) => {
  console.log(result);
});
```

FeatureExtractor(mobileNet model) Loaded

Load Dataset

If you load this sample classifer dataset. Try to make rock, paper, or scissor gestures to see if the classifier can class them.
If this sample dataset doesn't work well for you, you could train your own classifier,
and use the 'Save Dataset' button below to create your own myKNNDataset.json file,
and replace the myKNNDataset.json in this folder.

✊ Add an Example to Class Rock | Reset Class Rock | 12 Rock examples | Confidence in Rock is: 100 %

🖐 Add an Example to Class Paper | Reset Class Paper | 13 Paper examples | Confidence in Paper is: 0 %

✌ Add an Example to Class Scissor | Reset Class Scissor | 16 Scissor examples | Confidence in Scissor is: 0 %

Start predicting!

Clear all classes

KNN Classifier with mobileNet model labeled this as Class: Rock with a confidence of 100 %

Save Dataset

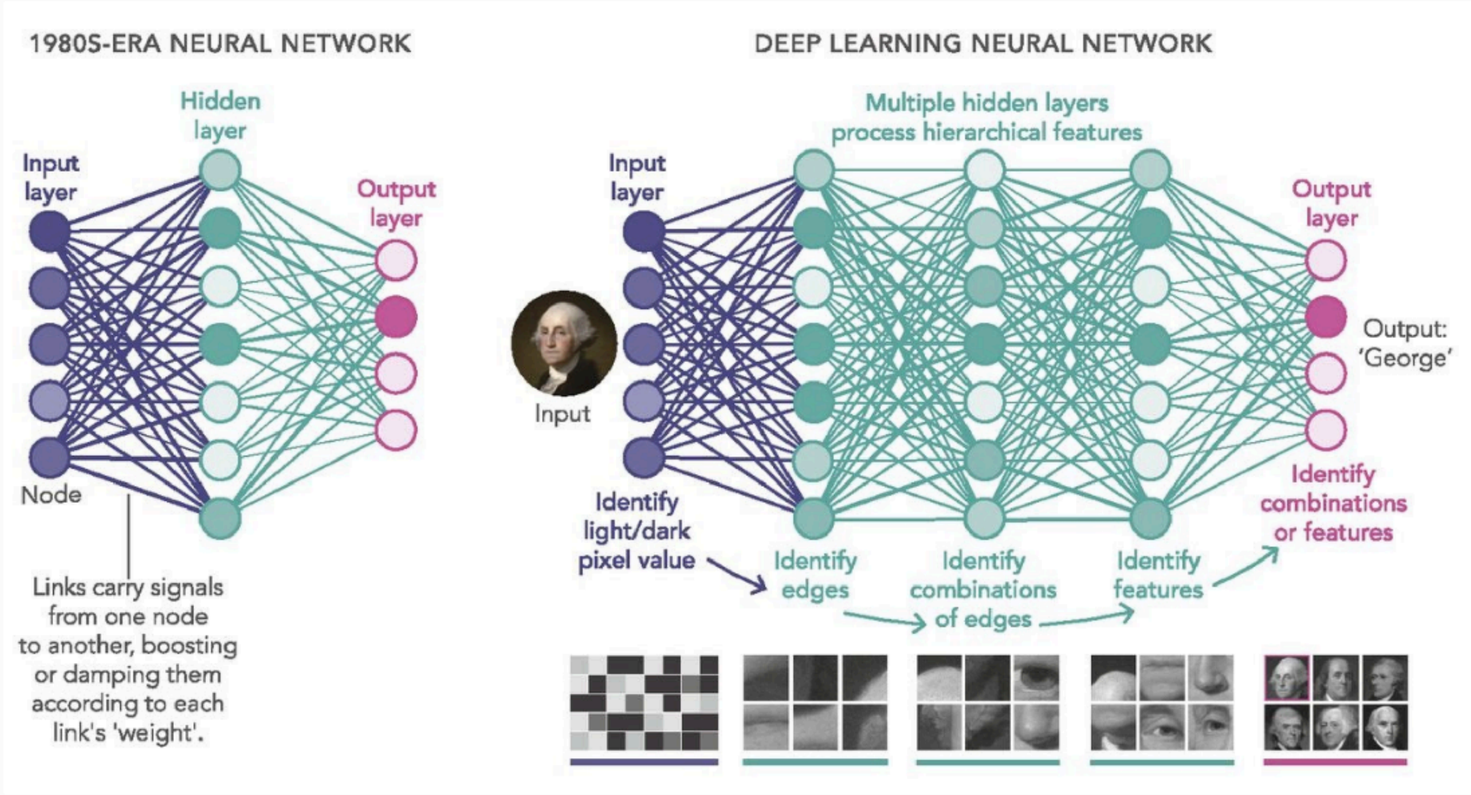
INTERACTION DESIGN ZHDK

BUILDING NN

MUI HS21

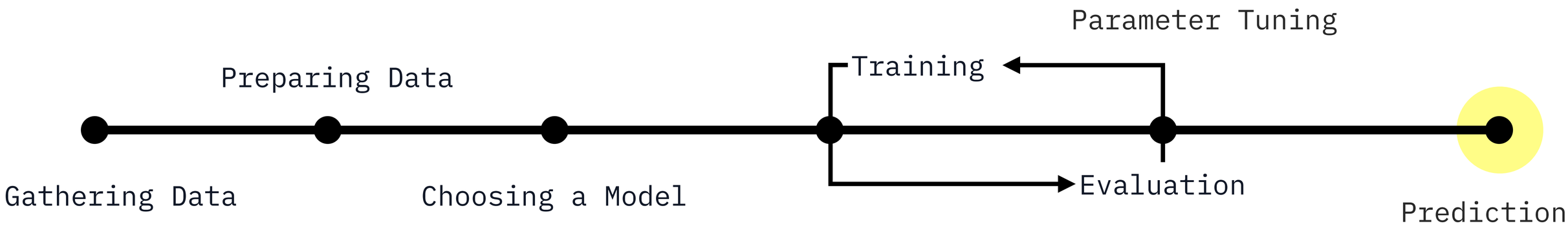
ARCHITECTURE

Neural Networks are the functional unit of Deep Learning and mimic the behavior of the human brain to solve complex data-driven problems.



https://www.v7labs.com/blog/neural-network-architectures-guide

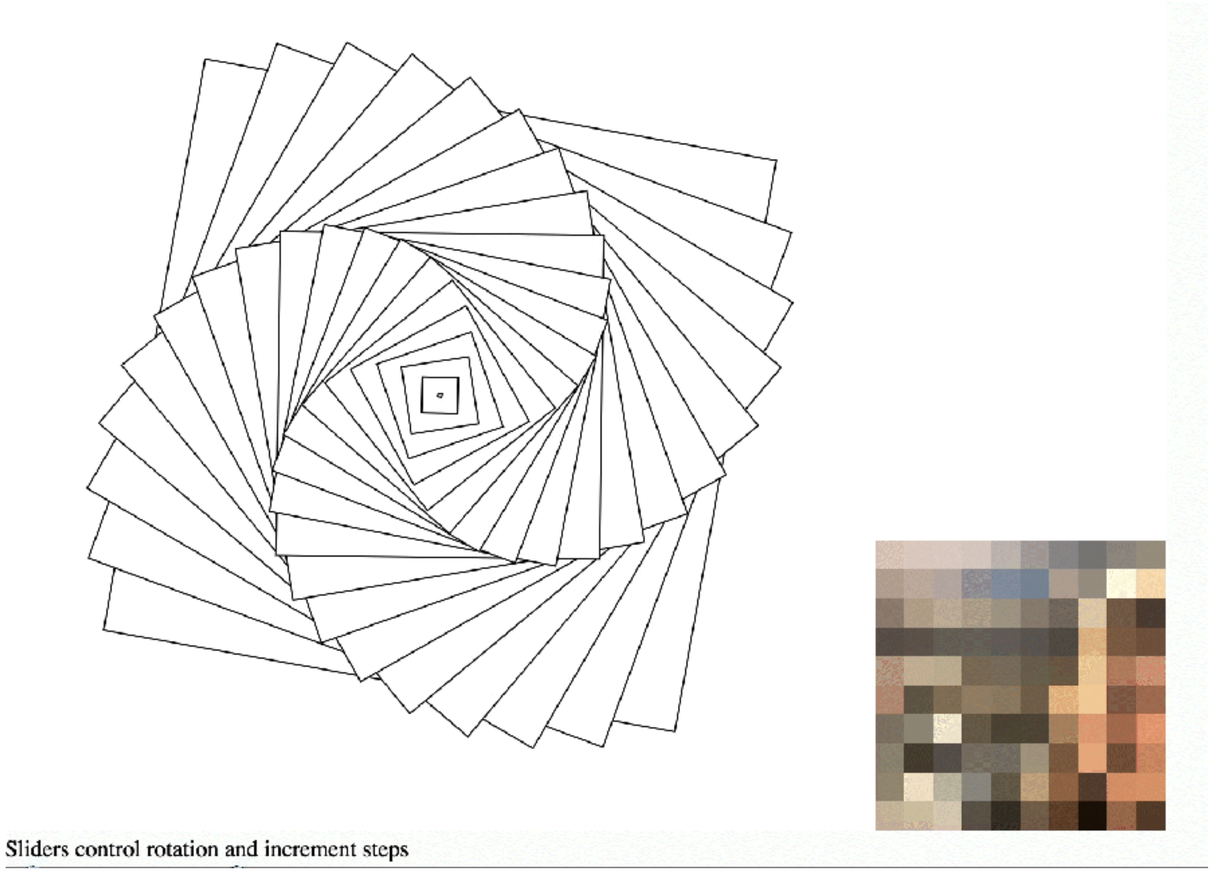
LEARNING STEPS



ML5.NEURALNETWORK

NeuralNetwork

Step 1: load data or create some data
Step 2: choose & initialize NN
Step 3: normalize data
Step 4: train the model
Step 5: use the trained model
Step 6: make a classification
Step 7: handle the results



NeuralNetwork

```
//Step 1 load data or create some data
const options = {
  inputs: [],
  outputs: [],
  dataUrl: null, // json or csv file
  modelUrl: null, //preloaded model
  layers: [], // custom layers
  task: null, // 'classification', ‘regression',
                'imageClassification'
  debug: true, // show training visualisation
  learningRate: 0.2,
  hiddenUnits: 16, //amount of hidden layers
};

// Step 2: initialize NN
const nn = ml5.neuralNetwork(options, dataLoaded);
```

NeuralNetwork

```
// Step 3: normalize data
function dataLoaded(){
  nn.normalizeData();
  trainModel();
}

// Step 4: train the model
function trainModel(){
  const trainingOptions = {
    epochs: 32, //amount of time dataset is being
    forward and backward through the NN
    batchSize: 12 //amount of parts data is divided into
  }
  nn.train(trainingOptions, finishedTraining);
}
```

NeuralNetwork

```
// Step 5: use the trained model
function finishedTraining(){
  classify();  //if classification is used
 //predict();   if regression is used

}

// Step 6: make a classification
function classify(){ //or predict()
  const input = {
    temperature: 5,
    snow_depth: 2,
    elevation: 30
  }
  nn.classify(input, handleResults);
  //nn.predict(input, handleResults)
}
```

NeuralNetwork

```
// Step 7: handle the results
function handleResults(error, result) {
    if(error){
      console.error(error);
      return;
    }
    console.log(result); // {label: ‘risk', confidence: 0.8};
}
```

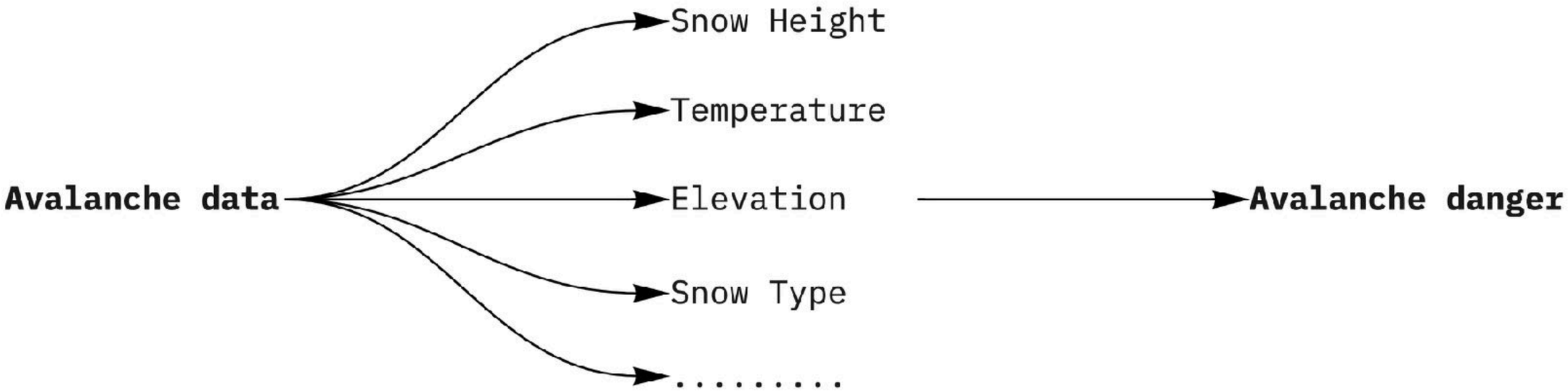
SAVE AND LOAD

NeuralNetwork

```
model.saveData('davos.csv'); //save data to .csv
model.save(); // save model

model.loadData('davos.csv', dataLoaded);
  const modelInfo = {
    model: 'model/model.json',
    metadata: 'model/model_meta.json',
    weights: 'model/model.weights.bin'
  }


function dataLoaded() {
  let data = model.data.data.raw;
  for (let i = 0; i < data.length; i++) {
    let inputs = data[I].xs; //
    let target = data[i].ys;
  }
```



https://www.envidat.ch/#/metadata/snow-avalanche-data-davos

Avalanche Prediction Template

https://editor.p5js.org/pzybinska/sketches/A5NrFgSdi