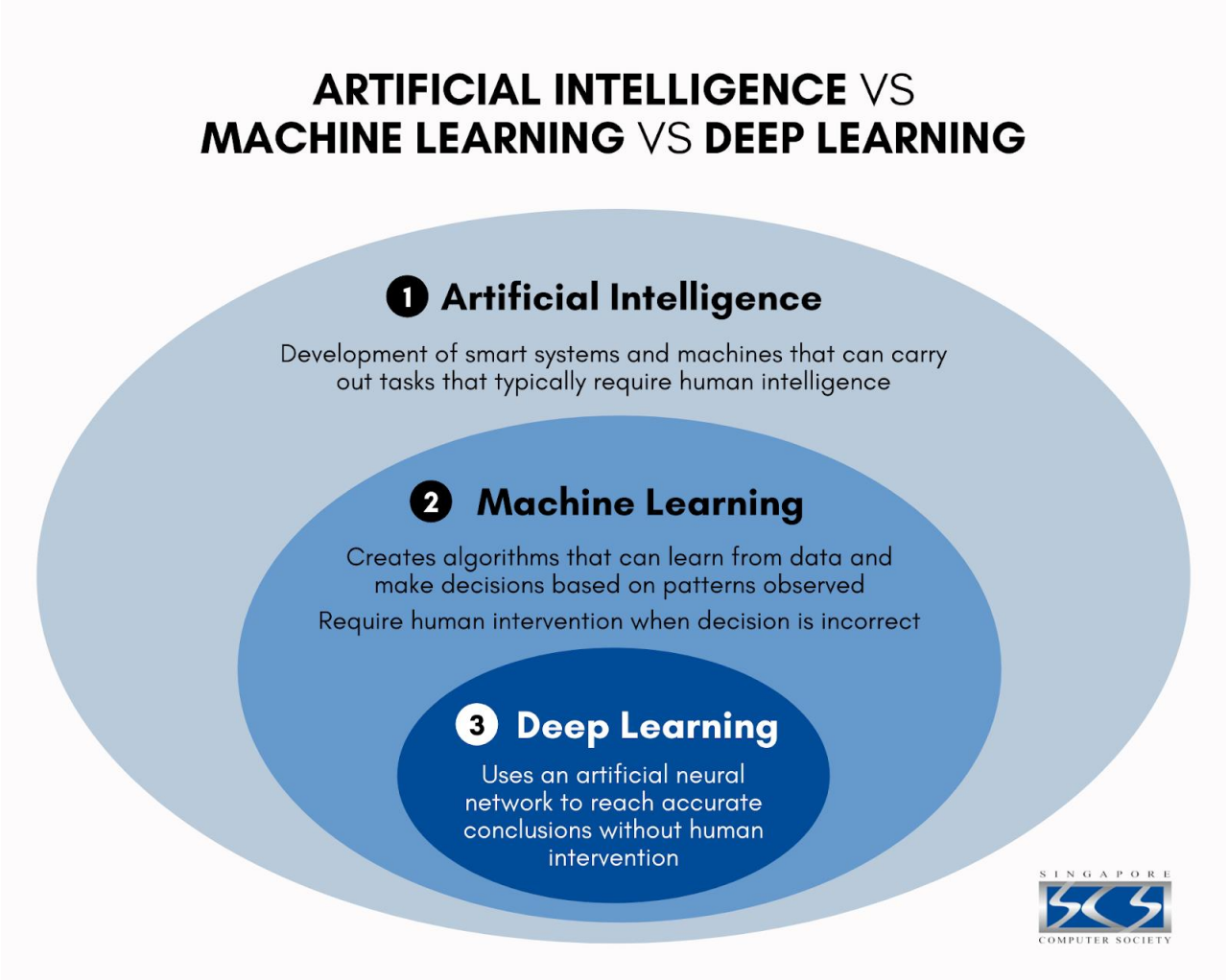
School of Mechanical Engineering
FACULTY OF ENGINEERING

UNIVERSITY OF LEEDS

MECH5170M
Connected and Autonomous Vehicles Systems

Machine learning algorithms

Kris Kubiak (k.kubiak@leeds.ac.uk)


ARTIFICIAL INTELLIGENCE VS
MACHINE LEARNING VS DEEP LEARNING
1 Artificial Intelligence
Development of smart systems and machines that can carry out tasks that typically require human intelligence
2 Machine Learning
Creates algorithms that can learn from data and make decisions based on patterns observed
Require human intervention when decision is incorrect
3 Deep Learning
Uses an artificial neural network to reach accurate conclusions without human intervention
SINGAPORE
COMPUTER SOCIETY

Machine Learning

Research Fields

Statistics / Statistical Learning

Data Mining

Pattern Recognition

Artificial Intelligence

Databases

Signal Processing

Machine Learning Algorithm Cheat Sheet

This cheat sheet helps you choose the best machine learning algorithm for your predictive analytics solution.
Your decision is driven by both the nature of your data and the goal you want to achieve with your data.

Microsoft Azure

What do you want to do?

Extract information from text

Text Analytics

Derives high-quality information from text

Answers questions like: What info is in this text?

- **Latent Dirichlet Allocation** ← Unsupervised topic modeling, group texts that are similar
- **Extract N-Gram Features from Text** ← Creates a dictionary of n-grams from a column of free text
- **Feature Hashing** ← Converts text data to integer encoded features using the Vowpal Wabbit library
- **Preprocess Text** ← Performs cleaning operations on text, like removal of stop-words, case normalization
- **Word2Vector** ← Converts words to values for use in NLP tasks, like recommender, named entity recognition, machine translation

Predict between several categories

Multiclass Classification

Answers complex questions with multiple possible answers

Answers questions like: Is this A or B or C or D?

- **Multiclass Logistic Regression** ← Fast training times, linear model
- **Multiclass Neural Network** ← Accuracy, long training times
- **Multiclass Decision Forest** ← Accuracy, fast training times
- **One-vs-All Multiclass** ← Depends on the two-class classifier
- **One-vs-One Multiclass** ← Depends on binary classifier, less sensitive to an imbalanced dataset with larger complexity
- **Multiclass Boosted Decision Tree** ← Non-parametric, fast training times and scalable

Predict between two categories

Two-Class Classification

Answers simple two-choice questions, like yes or no, true or false

Answers questions like: Is this A or B?

- **Two-Class Support Vector Machine** ← Under 100 features, linear model
- **Two-Class Averaged Perceptron** ← Fast training, linear model
- **Two-Class Decision Forest** ← Accurate, fast training
- **Two-Class Logistic Regression** ← Fast training, linear model
- **Two-Class Boosted Decision Tree** ← Accurate, fast training, large memory footprint
- **Two-Class Neural Network** ← Accurate, long training times

Classify images

Image Classification

Classifies images with popular networks

Answers questions like: What does this image represent?

- **ResNet** ← Modern deep learning neural network
- **DenseNet** ← Modern deep learning neural network

Generate recommendations

Recommenders

Predicts what someone will be interested in

Answers the question: What will they be interested in?

- **Use the Train Wide & Deep Recommender module** ← Hybrid recommender, both collaborative filtering and content-based approach
- **SVD Recommender** ← Collaborative filtering, better performance with lower cost by reducing dimensionality

Predict values

Regression

Makes forecasts by estimating the relationship between values

Answers questions like: How much or how many?

- **Fast Forest Quantile Regression** ← Predicts a distribution
- **Poisson Regression** ← Predicts event counts
- **Linear Regression** ← Fast training, linear model
- **Bayesian Linear Regression** ← Linear model, small data sets
- **Decision Forest Regression** ← Accurate, fast training times
- **Neural Network Regression** ← Accurate, long training times
- **Boosted Decision Tree Regression** ← Accurate, fast training times, large memory footprint

Discover structure

Clustering

Separates similar data points into intuitive groups

Answers questions like: How is this organized?

- **K-Means** ← Unsupervised learning

Find unusual occurrences

Anomaly Detection

Identifies and predicts rare or unusual data points

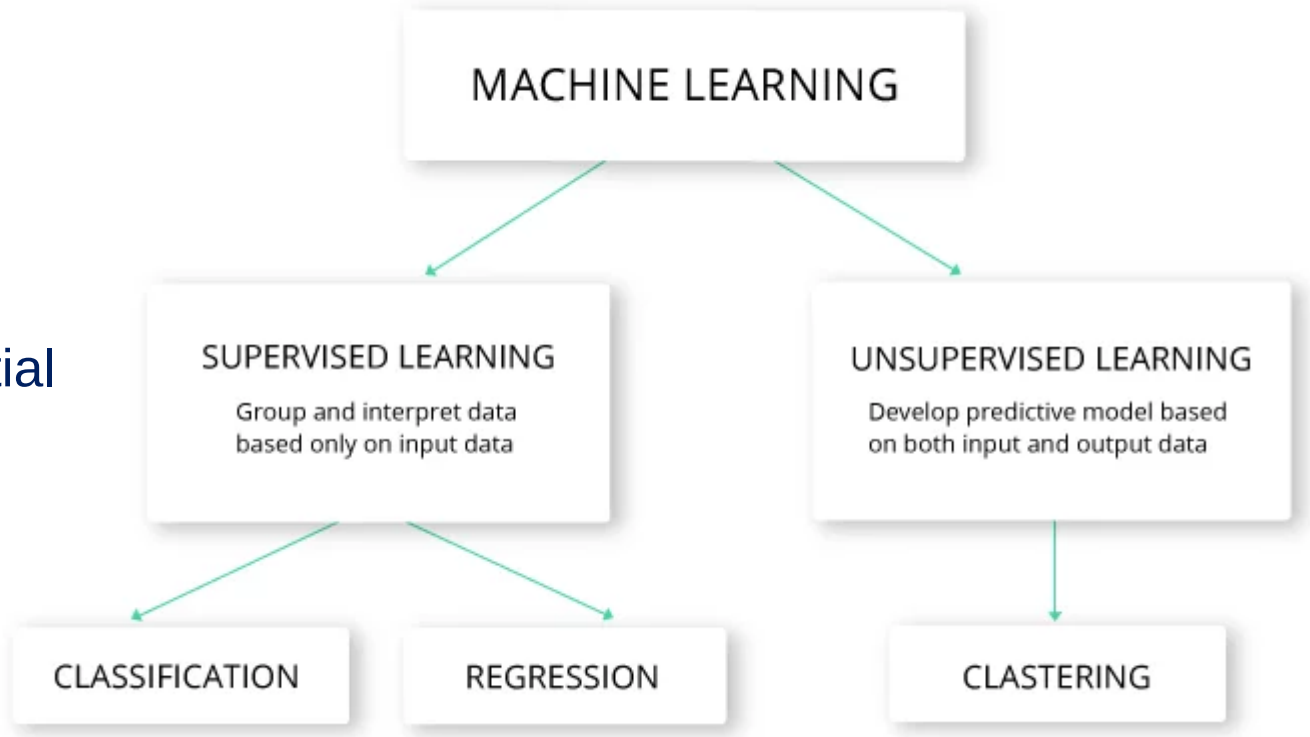
Answers the question: Is this weird?

- **One Class SVM** ← Under 100 features, aggressive boundary
- **PCA-Based Anomaly Detection** ← Fast training times

 Share this poster: aka.ms/mlcheatsheet

ML in Autonomous Vehicles

Training is essential
No self-learning



Self-learning possible

ML in Autonomous Vehicles

Object Detection	Clustering
Object Classification	Multiclass Classification
Distance, Speed, Direction Estimation	Regression - value prediction
Path Finding	Optimisation Algorithms
Trajectory planning	Collision avoidance (prediction)
Motor/Battery Monitoring	Regression
Anomaly Detection	Pattern Recognition

Model Building Process

Data Splitting

- Allocate data to different tasks
  - Model training
  - Performance evaluation
- Define Training, Validation and Test sets

Feature Selection (Review the decision made previously)

Estimating Performance

- Visualization of results – discover interesting areas of the problem space
- Statistics and performance measures

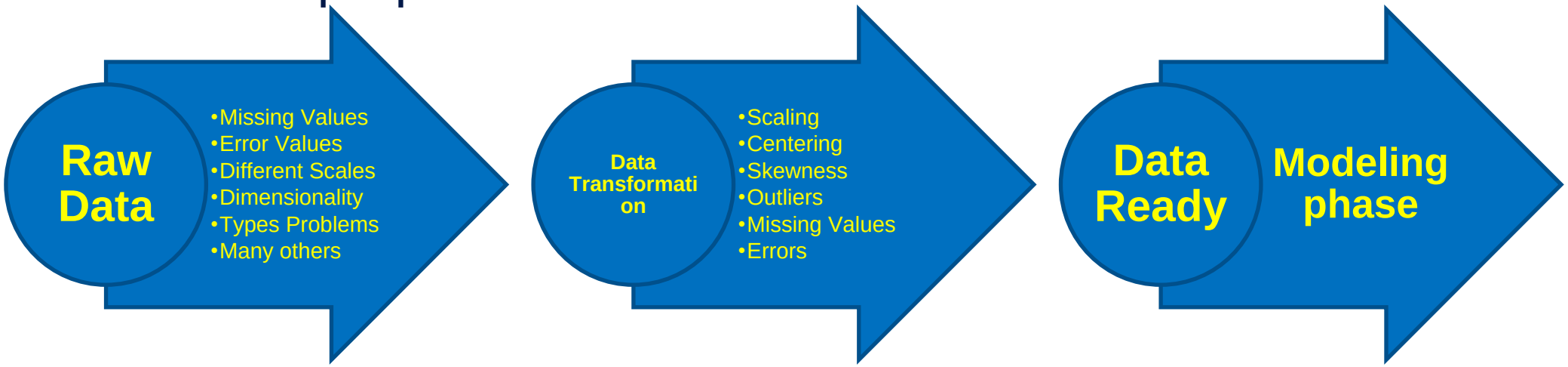
Evaluation and Model selection

- No priory assumptions should be made
- Avoid use of favorite models, evaluate and select the best performing one

ML Data Preparation

UNIVERSITY OF LEEDS

- Needed for several reasons
  - Some Models have strict data requirements
    - Scale of the data, data point intervals, etc.
  - Some characteristics of the data may impact dramatically on the model performance
- Time on data preparation should not be underestimated



Supervised Learning

- For every example in the data there is always a predefined outcome
- Models the relations between a set of descriptive features and a target (Fits data to a function)
- 2 groups of problems:
  - Classification
  - Regression

Classification

- Predicts which class a given sample of data (sample of descriptive features) is part of (**discrete value**)

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa

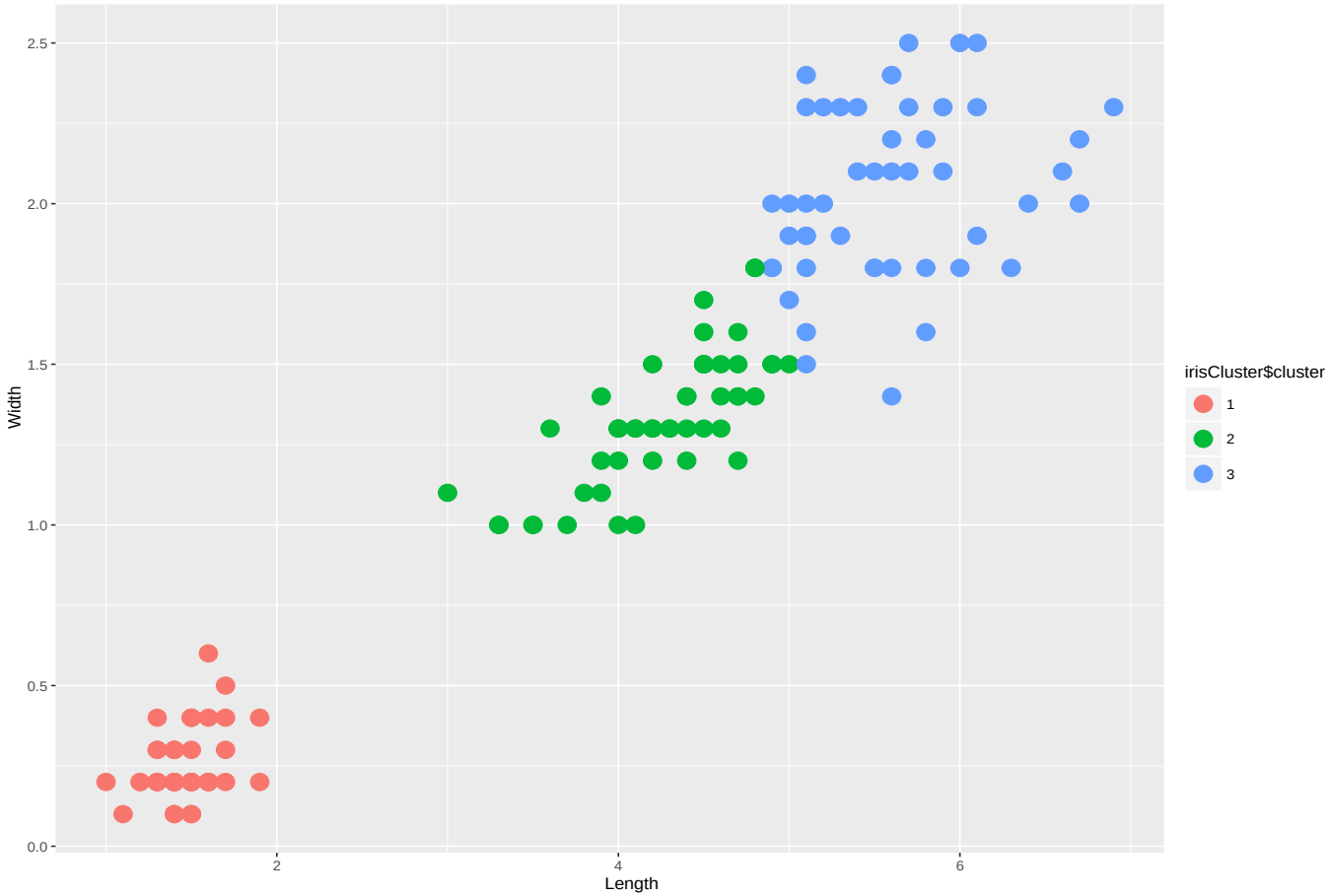
Regression

- Predicts continuous values

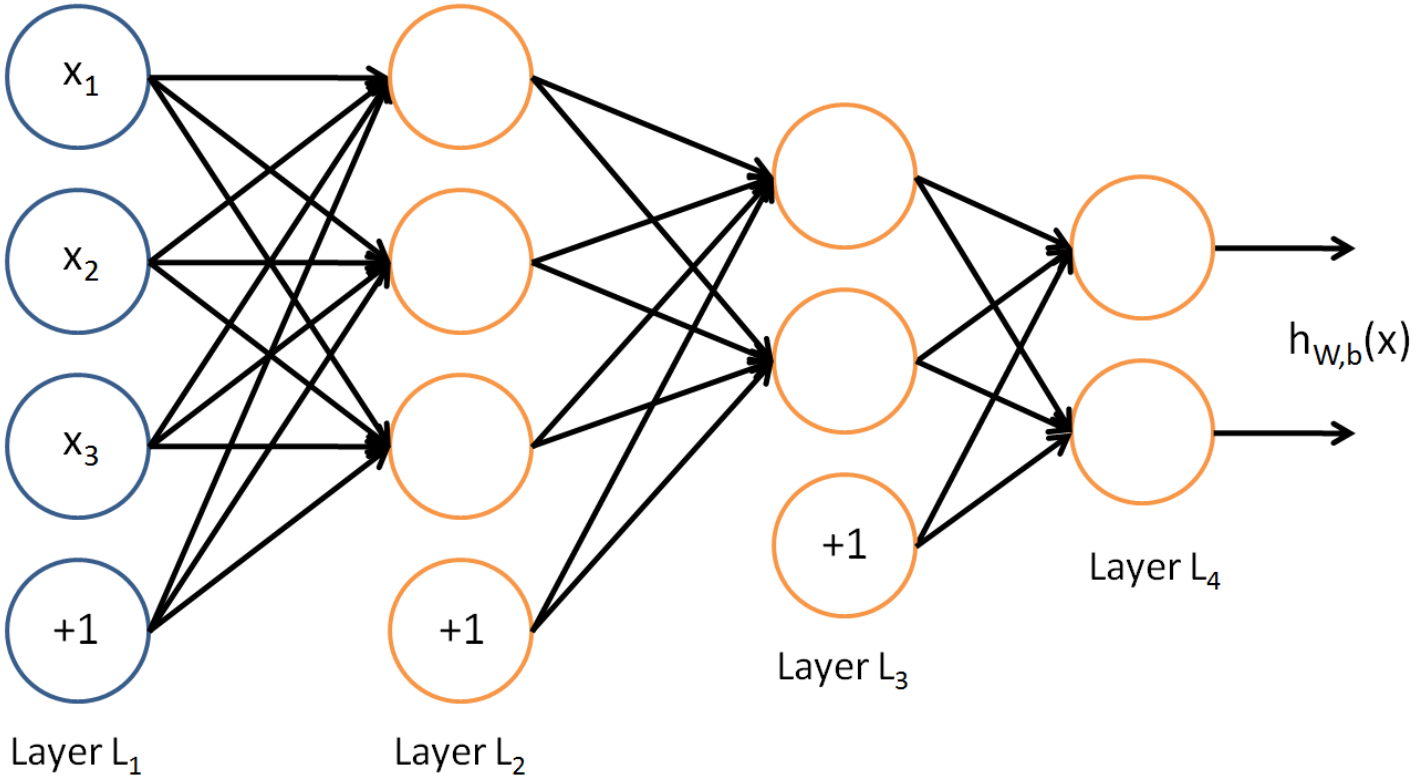
Unsupervised Learning

- There are not predefined and known set of outcomes
- Look for hidden patterns and relations in the data
- A typical example: Clustering

	Length	Width	Length	Width
1	5.1	3.5	1.4	0.2
2	4.9	3.0	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
5	5.0	3.6	1.4	0.2
6	5.4	3.9	1.7	0.4
7	4.6	3.4	1.4	0.3
8	5.0	3.4	1.5	0.2
9	4.4	2.9	1.4	0.2
10	4.9	3.1	1.5	0.1



Multilayer network



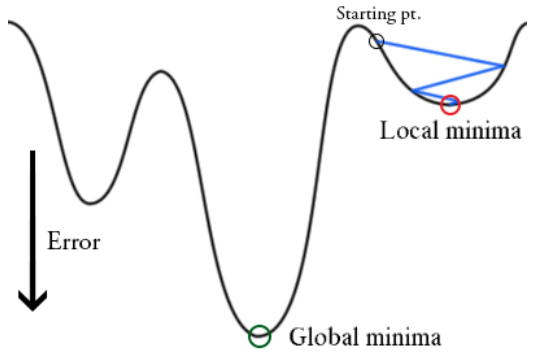
How to determine weights?

Training: backpropagation

Initialize weights "randomly"

For all training epochs

- for all input-output in training set
  - using input, compute output (forward)
  - compare computed output with training output
  - adapt weights (backward) to improve output
- if accuracy is good enough, stop



The Learning Problem

Learning from data is used in situations where we don’t have any analytical solution, but we do have data that we can use to construct an empirical solution

The basic premise of learning from data is the use of a set of observations to uncover an underlying process.

The Learning Problem

Suppose we observe the output space Y_i and the input space

$$X_i = (X_{i1}, \ldots, X_{ip}) \; \forall \; i = 1, \ldots, n$$

We believe that there is a *relationship* between *Y* and at least one of the *X*'s.

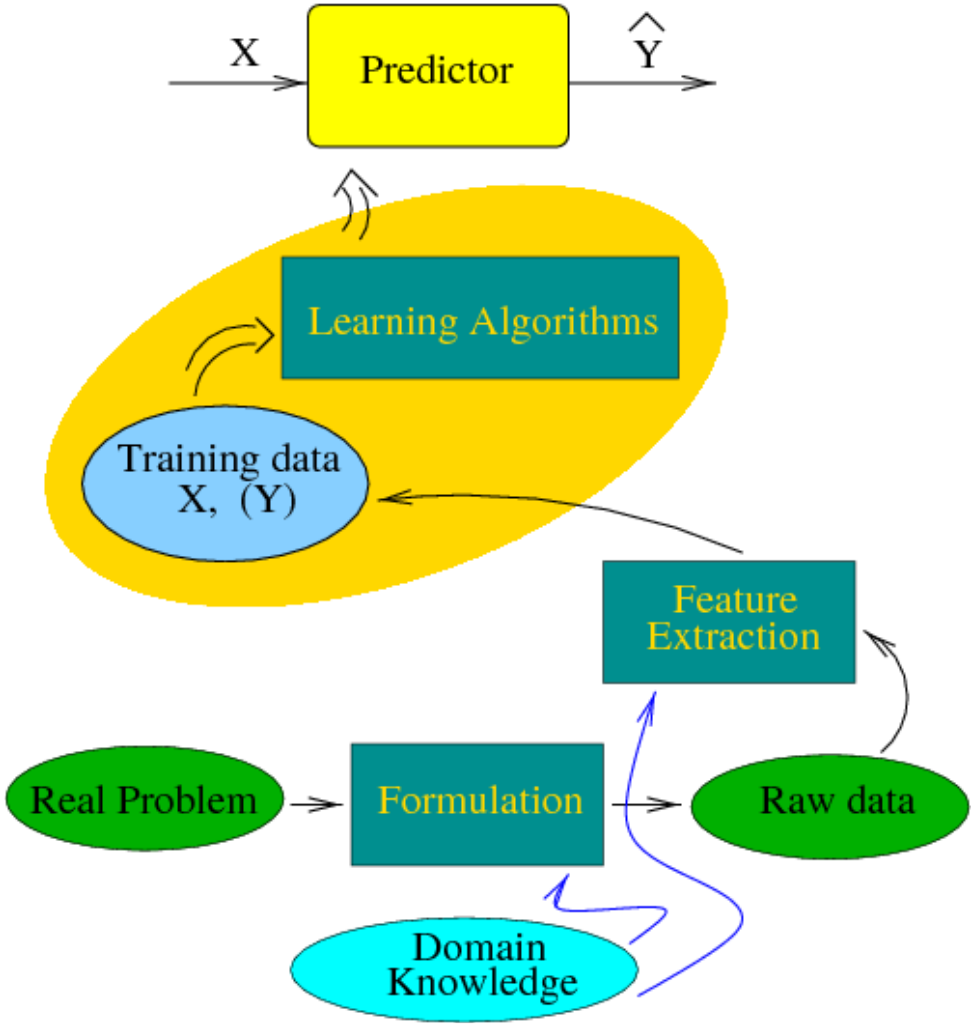
We can model the relationship as:

$$Y_i = f(\boldsymbol{X_i}) + \varepsilon_i$$

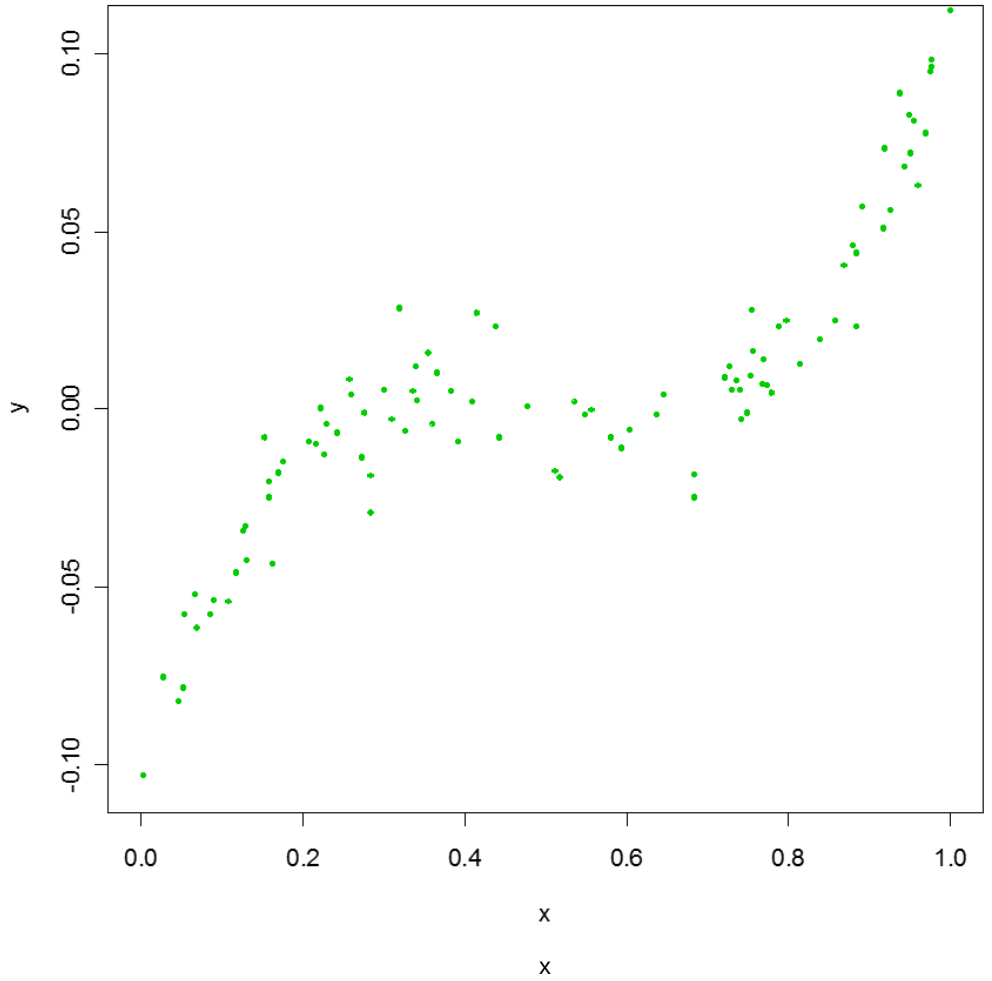
where *f* is an unknown function and ε is a random error (noise) term, independent of $\boldsymbol{X}$ with mean zero.

The Learning Problem

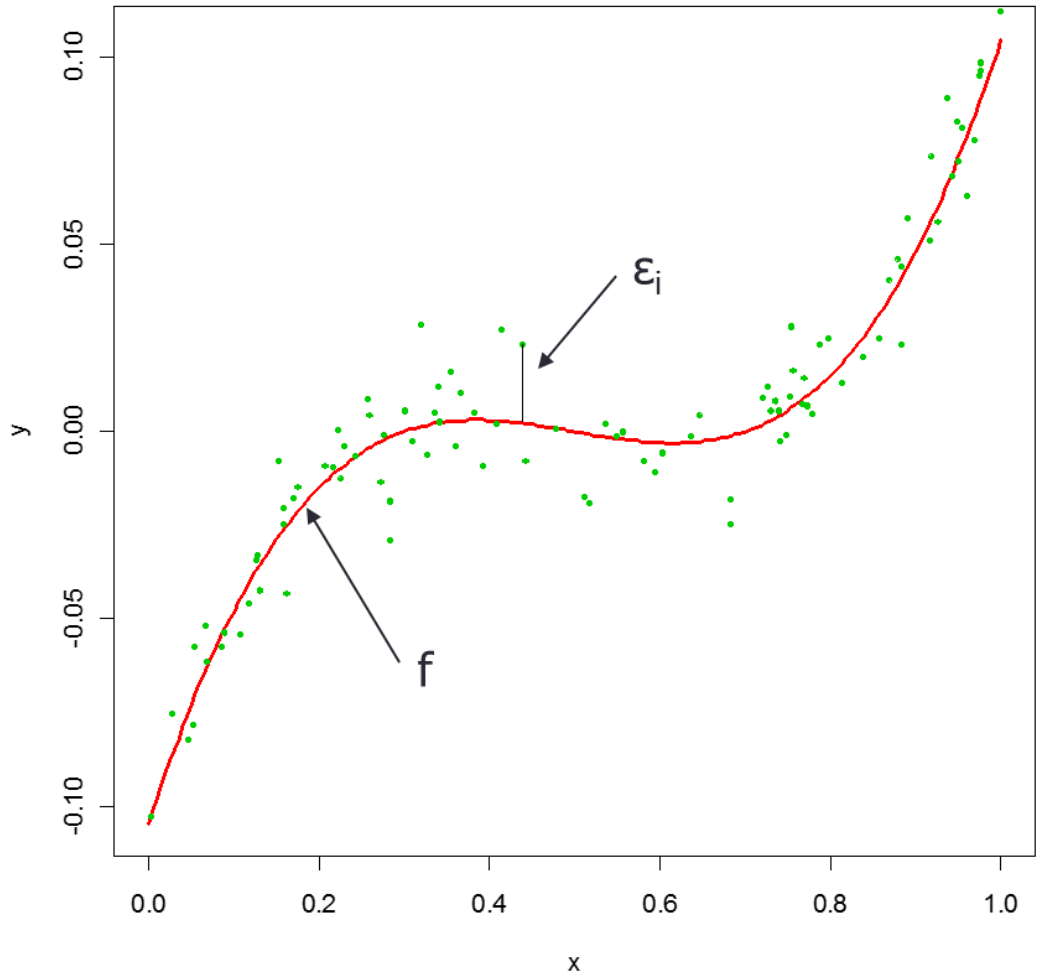
UNIVERSITY OF LEEDS



The Learning Problem: Example



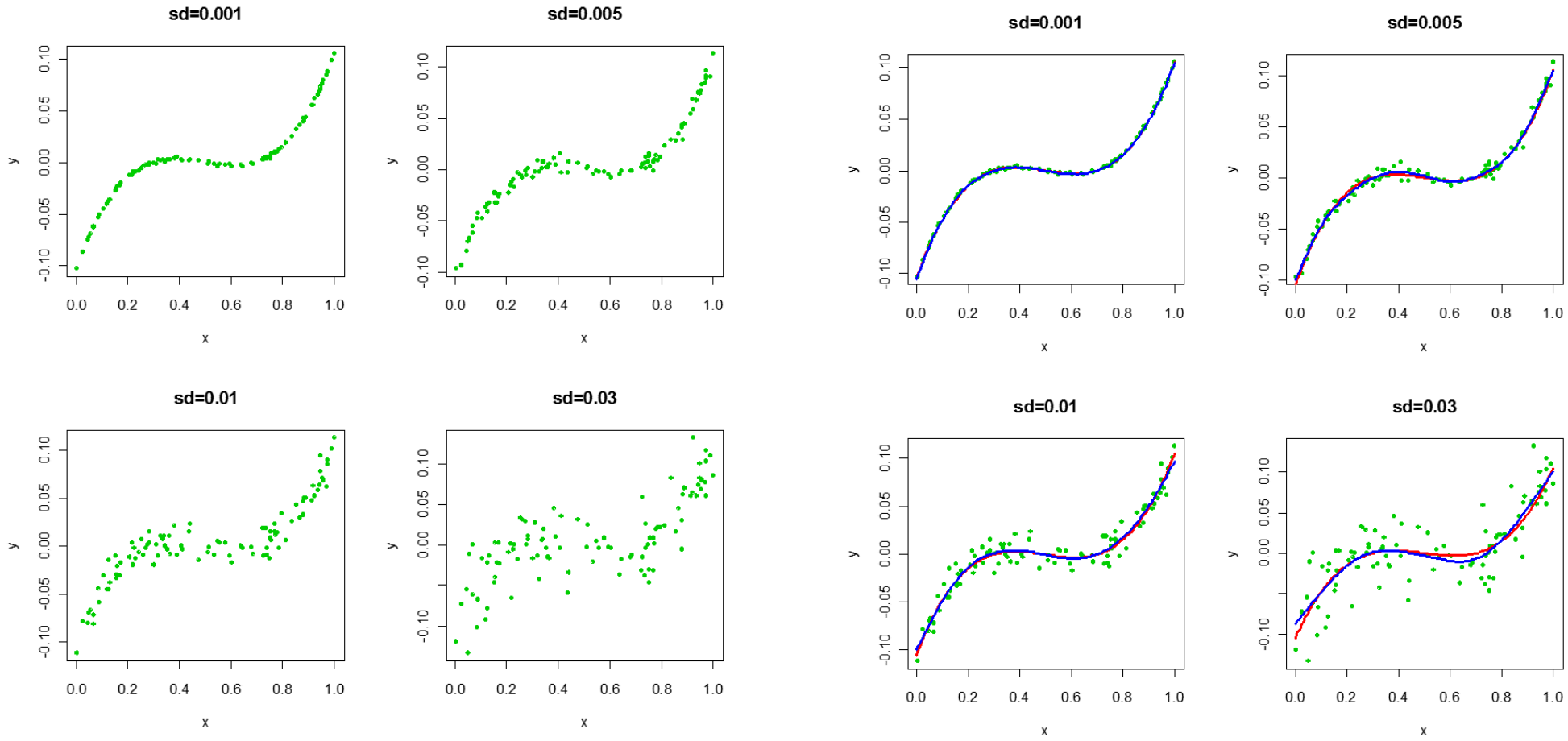
The Learning Problem: Example



The Learning Problem: Example

UNIVERSITY OF LEEDS

Different estimates for the target function *f* that depend on the standard deviation of the ε's



Why do we estimate *f* ?

We use modern machine learning methods to estimate *f* by *learning* from the data.

The target function *f* is unknown.

We estimate *f* for two key purposes:

- Prediction
- Inference

Prediction

UNIVERSITY OF LEEDS

By producing a good estimate for *f* where the variance of ε is not too large, then we can make accurate predictions for the response variable, *Y*, based on a new value of $\boldsymbol{X}$.

We can predict *Y* using $\hat{Y} = \hat{f}(\boldsymbol{X})$

where $\hat{f}$ represents our estimate for *f*, and $\hat{Y}$ represents the resulting prediction for *Y*.

Prediction

The accuracy of $\hat{Y}$ as a prediction for Y depends on:

- Reducible error
- Irreducible error

Note that $\hat{f}$ will not be a perfect estimate for f; this inaccuracy introduces error.

Prediction

This error is ***reducible*** because we can potentially improve the accuracy of the estimated (i.e. hypothesis) function $\hat{f}$ by using the most appropriate learning technique to estimate the target function *f*.

Even if we could perfectly estimate *f*, there is still **variability associated with *ε*** that affects the accuracy of predictions = ***irreducible*** error.

Prediction

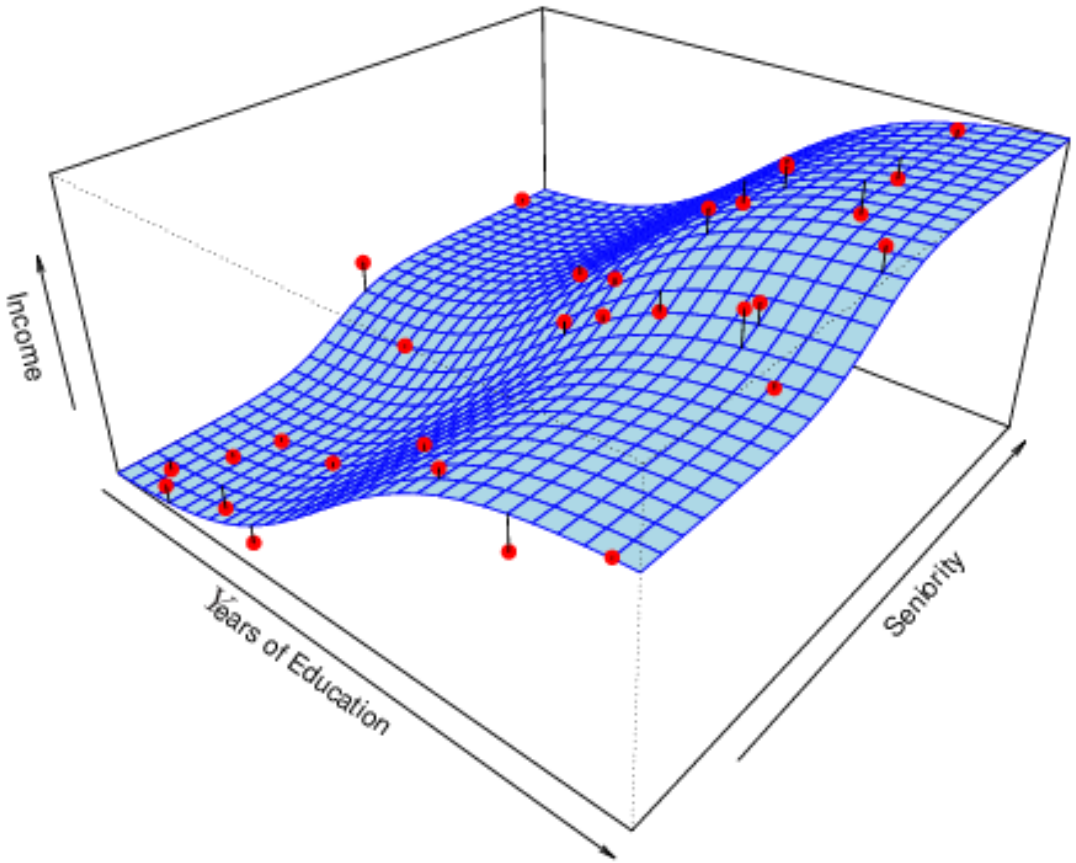
Average of the squared difference between the predicted and actual value of *Y*.

Var(*ε*) represents the *variance* associated with *ε*.

$$E[(Y - \hat{f}(X))^2 | X = x] = \underbrace{[f(x) - \hat{f}(x)]^2}_{Reducible} + \underbrace{\mathrm{Var}(\epsilon)}_{Irreducible}$$

Our aim is to minimize the reducible error!!

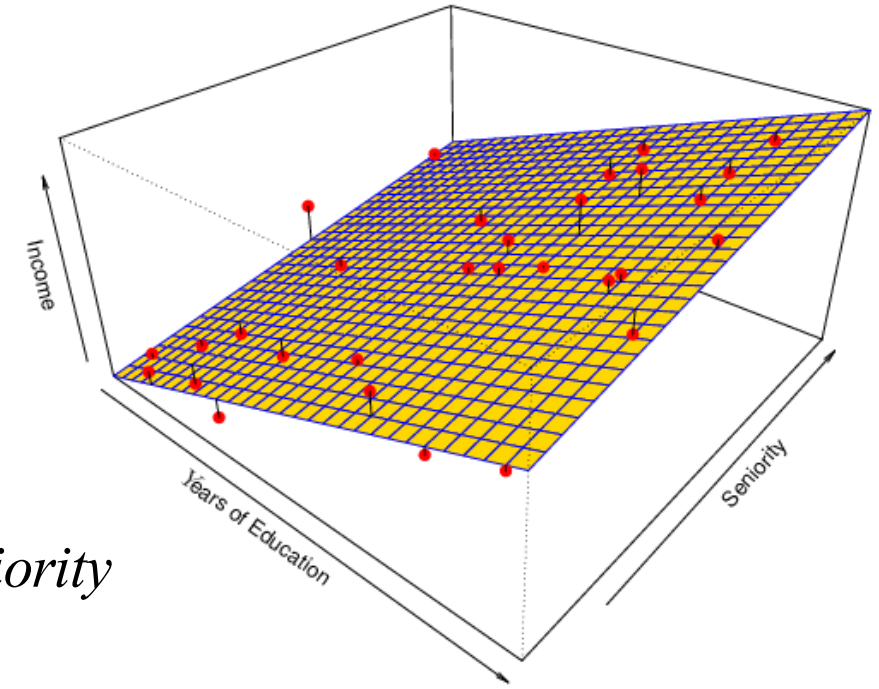
Example: Income vs. Education Seniority



Example: OLS Regression Estimate

UNIVERSITY OF LEEDS

Even if the standard deviation is low, we will still get a bad answer if we use the incorrect model.



$$f = b_0 + b_1 \times Education + b_2 \times Seniority$$

Machine Learning Conclusions

- Machine learning
  - Supervised (Training before deployment)
  - Unsupervised (Self-learning capabilities)

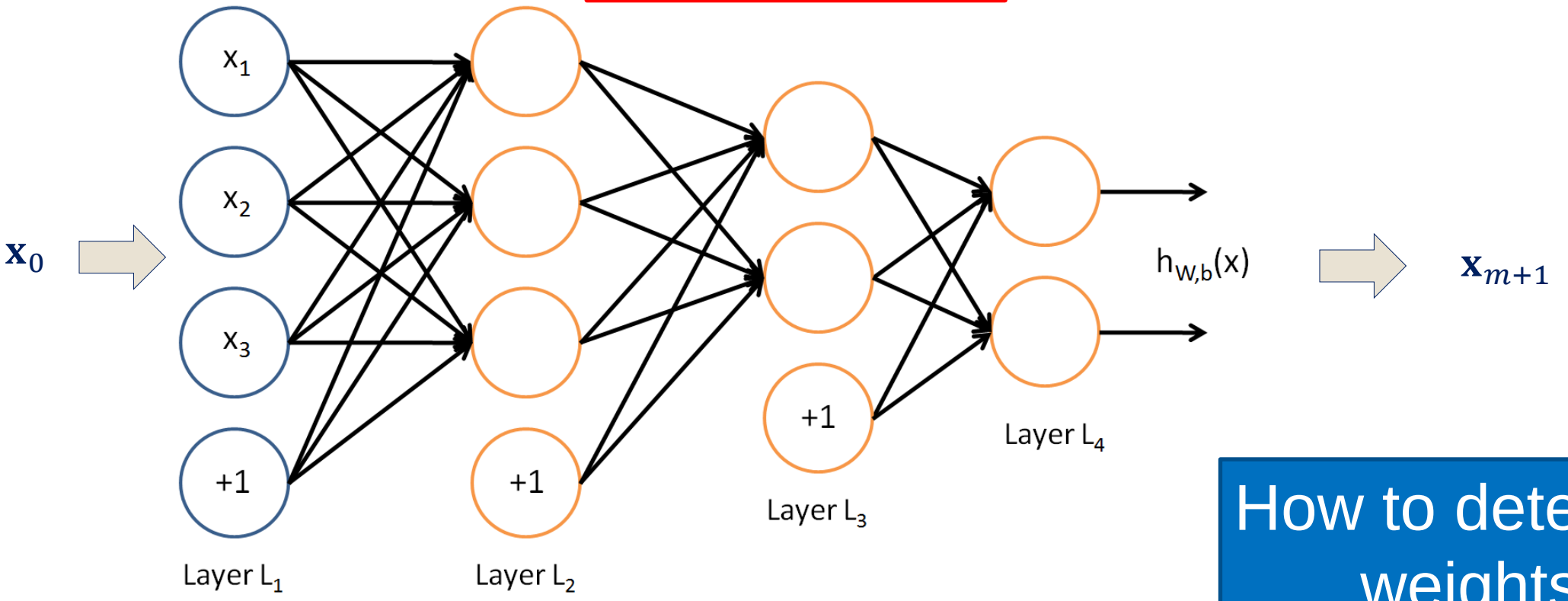
- Clustering
- Classification
- Regression, Prediction

Neural Networks

Multilayer network

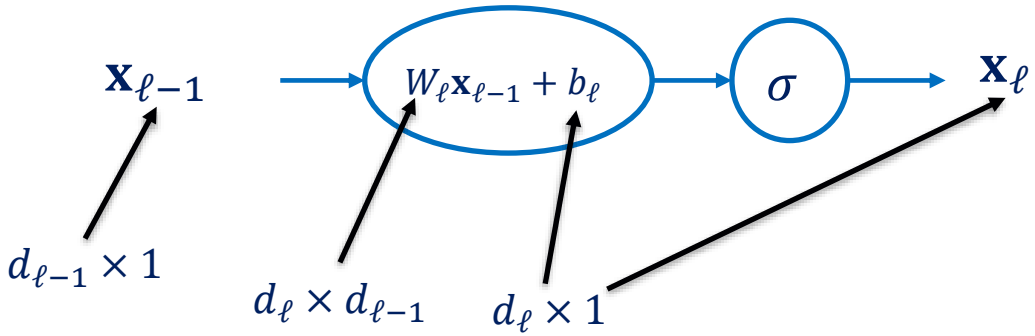
UNIVERSITY OF LEEDS

W_ℓ the weight matrix



How to determine weights?

Basics of Neural networks



Compact representation of the ℓ^{th} layer of a neural network

A feedforward neural network with m hidden layers is defined as follows:

- $\mathbf{x}_0$ = Input to the network
- $\mathbf{x}_{m+1}$ = Output of the network
- $\mathbf{x}_\ell$ = Output of the ℓ^{th} layer
- And, $\mathbf{x}_\ell = f_\ell(\mathbf{x}_{\ell-1})$
- Where, $f_\ell = \sigma_\ell(W_\ell \mathbf{x}_{\ell-1} + b_\ell)$
- σ_ℓ is known as the activation function, W_ℓ the weight matrix and, b_ℓ the bias term of ℓ^{th} layer

Basics of Training a NN

Compute “how off” you are from the given example

$$J(W, b; x, y) = \frac{1}{2} \left\| h_{W,b}(x) - y \right\|^2$$

J is the cost function for a given set of weights W, biases b and a training example x and y

Add a regularisation term (to avoid overfitting). This is a term proportional to the sum of squares of all elements in the weight matrics

Use gradient descent to minimise cost

- update each “variable” (weights/biases) : $W_{i,j} = W_{i,j} - \alpha \frac{\partial}{\partial W_{i,j}} J(W, b)$
- Here, α is a learning rate, and the partial derivative is computed by back-propagation (chain rule applied to the function represented by the NN)

Convolutional Neural Networks

Inspired by visual cortex in animals

Learns image filters that were previously hand-engineered

Basic intuitions for CNNs:

- Images are too large to be monolithically processed by a feedforward neural network (1000x1000 image = 10^6 inputs, which means the weight matrix for the second layer is proportional to at least $10^{6!}$)
- Data in an image is spatially correlated

CNN divided into several layers with different purposes

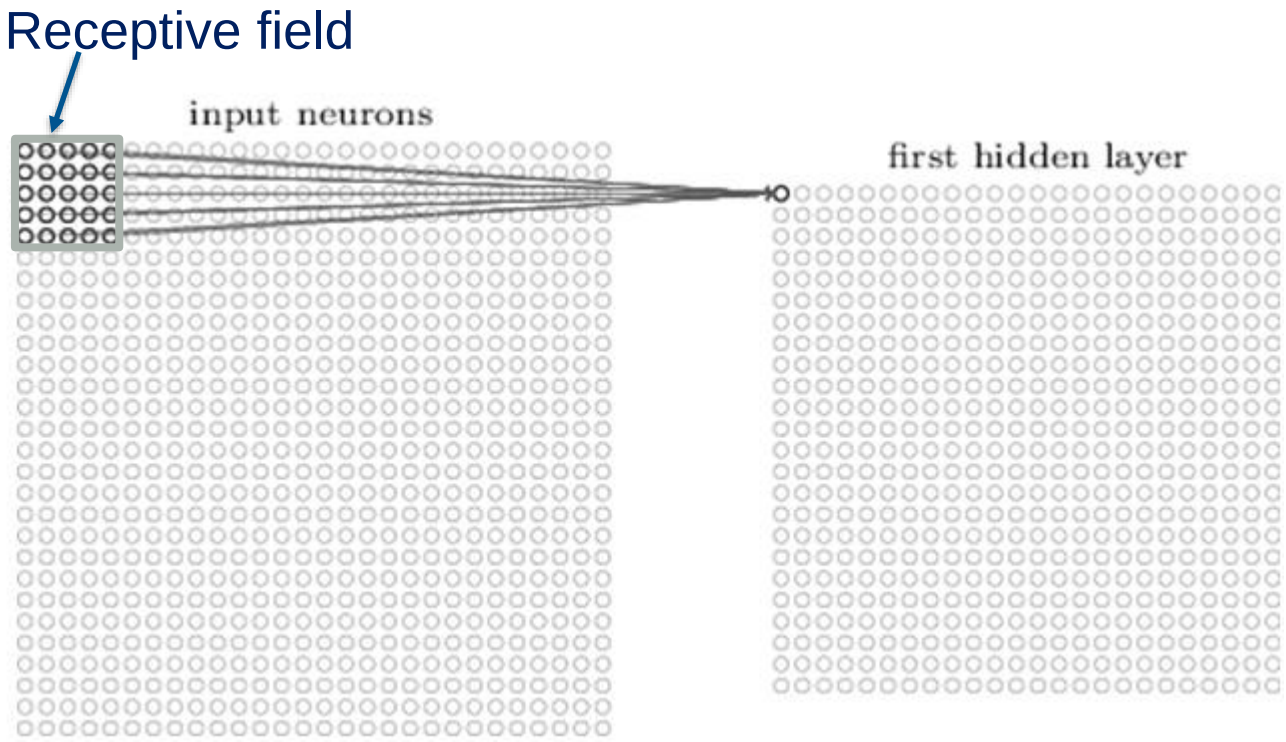
Convolutional layer

First layer is a convolutional layer

Convolutional layer contains neurons associated with sub-regions of the original image

Each sub-region is called receptive field

Convolves the weights of the convolutional layer with each cell in receptive field to obtain *activation map* or *feature map*



“Convolution”

Convolution of a 2-D image I with a 2-D kernel K defined as:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

Most neural network libraries do not use convolution, but instead implement **cross-correlation**, i.e.

$$S(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n)$$

The kernel function K usually defines the receptive field

Purpose of convolutional layer

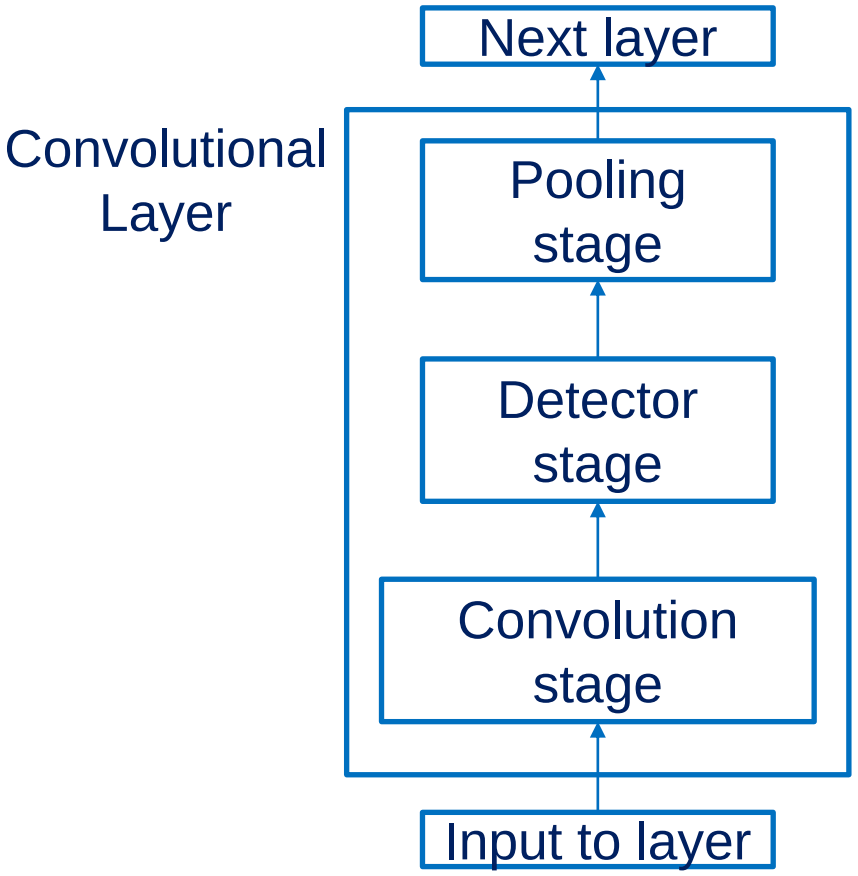
Convolutional layer **applies a *filter* to the pixels** within its **receptive field**

This allows **identifying low-level features** (curves, straight lines, etc.)

The outputs of the first convolutional layer can be thought of as having a "**high value**" when a particular **feature is detected** and a "**low value**" **otherwise**

Second convolutional layer allows **learning higher-level features** (e.g. semi-circles, angles etc.)

- Second convolutional layer has a bigger receptive field (as it is able to simultaneously correlate over outputs of first layer)
- By "convolving" over the feature map, **output of second layer tries to connect higher level features**



CNN architecture

- **Small number of layers**, where each layer contains stages
- **Convolution stage** performs convolution
- **Detector stage** uses a nonlinearity such as a rectified linear unit, i.e. $\max(x, 0)$
- **Pooling stage** performs a suitable pooling operation

Pooling stage

Pooling function **replaces the output** of a layer at a certain location **with a summary statistic** of the nearby outputs

For example: **max pooling** reports **maximum output** within a rectangular neighbourhood

Other pooling functions include **average**, **L^2 norm** (Euclidean distance), **weighted average** etc.

Pooling helps representation approximately **invariant to small translations**

By **pooling** over outputs of **different convolutions**, features can learn which transformations to become invariant to (e.g. rotation etc.)

Fully connected layers

CNNs may have some **fully connected layers before the final output**

These layers allows performing **higher-level reasoning** over different **features learned by** the **previous** convolutional **layers**

Various kind of **convolution** functions, **pooling** functions and **detection** functions are possible, giving rise to many different flavors

Number of convolutional **layers** can be varied depending on complexity of features to be learned

YOLO algorithm (You Only Look Once)

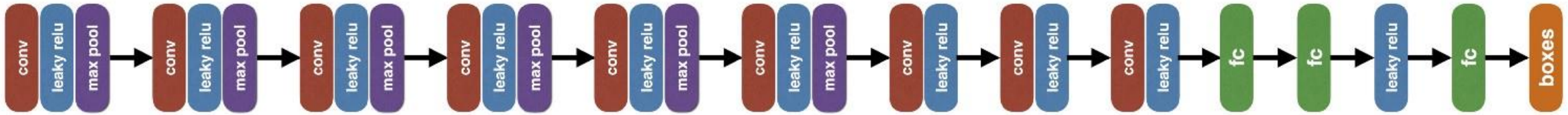
YOLO is one of the fastest real-time detection algorithms

R-CNN etc. are algorithms that leverage classifiers and localisers to perform detection

YOLO applies a single neural network to the entire image

Network divides image into regions and predicts bounding boxes and probabilities for each region

Bounding boxes are weighted by predicted probabilities



Annotations

UNIVERSITY OF LEEDS

- category
- attribute
- visibility
- instance
- sensor
- calibrated_sensor
- ego_pose
- log
- scene
- sample
- sample_data
- sample_annotation
- map

animal
debris
pushable_pullable
bicycle_rack
ambulance
police
barrier
bicycle
bus.bendy
bus.rigid
car
construction
motorcycle
adult
child
construction_worker
police_officer
personal_mobility
stroller
wheelchair
trafficcone
trailer
truck

type	Car, Van, Truck, Pedestrian, Person_sitting, Cyclist, Tram, Misc or DontCare
truncated	0 to 1, where truncated refers to the object leaving image boundaries
occuluded	0 = fully visible, 1 = partly occluded, 2 = largely occluded, 3 = unknown
alpha	Observation angle of object, ranging [-pi..pi]
bbox	2D bounding box of object in the image
dimensions	3D object dimensions: height, width, length
location	3D object location x,y,z in camera coordinate
rotation_y	Rotation ry around Y-axis in camera coordinates [-pi..pi]

Conclusions

Machine learning:

- Supervised (Training before deployment)
- Unsupervised (Self-learning capabilities)

Neural Network - Main steps:

- Object recognition
- Object perception
- Objects annotations

Image recognition:

- YOLO is often used for the image processing
- Bounding boxes are used, and annotated

ANY QUESTIONS ???