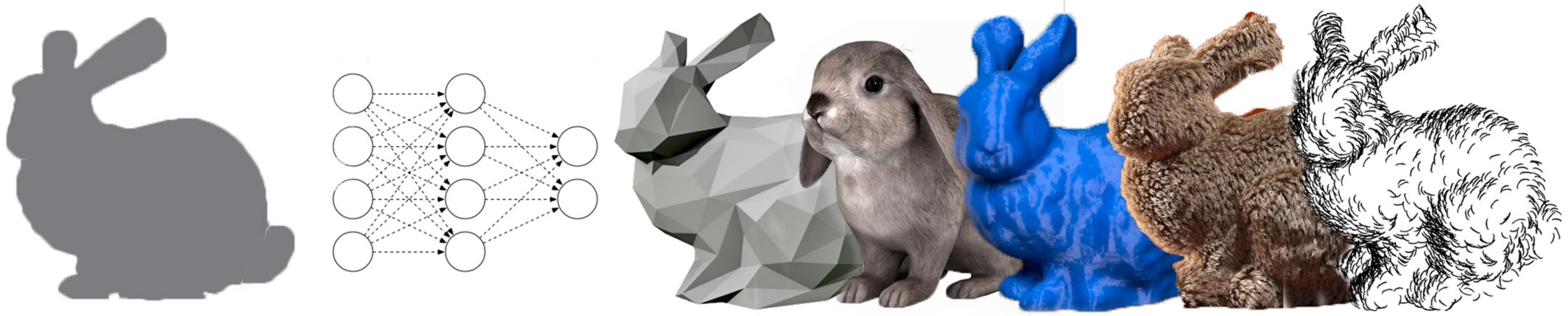
LOS ANGELES • 28 JULY – 1 AUGUST



Deep Learning for Computer Graphics

Niloy Mitra	Iasonas Kokkinos	Paul Guerrero	Vladimir Kim	Nils Thuerey	Leonidas Guibas
UCL/Adobe	UCL/Ariel AI	UCL/Adobe	Adobe	TU Munich	Stanford University/FAIR

Course Organizers

Niloy Mitra

Course Organizers

Niloy Mitra

Iasonas Kokkinos

Course Organizers

Niloy Mitra

Iasonas Kokkinos

Paul Guerrero

Course Organizers

Niloy Mitra

Iasonas Kokkinos

Paul Guerrero

Vladimir Kim

Course Organizers

Niloy Mitra

Iasonas Kokkinos

Paul Guerrero

Vladimir Kim

Nils Thuerey

Course Organizers

Niloy Mitra

Iasonas Kokkinos

Paul Guerrero

Vladimir Kim

Nils Thuerey

Leonidas Guibas

Course Organizers

Niloy Mitra

Iasonas Kokkinos

Paul Guerrero

Vladimir Kim

Nils Thuerey

Leonidas Guibas

Timetable

		Niloy	Iasonas	Paul	Nils	Leonidas
Introduction	9:00	X				
Neural Network Basics	~9:15		X			
Supervised Learning in CG	~9:50	X				
Unsupervised Learning in CG	~10:20			X		
Learning on Unstructured Data	~10:55					X
Learning for Simulation/Animation	~11:35				X	
Discussion	12:05	X	X	X	X	X

Course Objectives

Course Objectives

- Provide an overview of the popular **ML algorithms** used in CG
- Provide a quick overview of **theory** and **CG applications**

Course Objectives

- Provide an overview of the popular **ML algorithms** used in CG
- Provide a quick overview of **theory** and **CG applications**
  - Many extra slides in the course notes + example code
- Summarize progress in the last 3-5 years

Course Objectives

- Provide an overview of the popular **ML algorithms** used in CG
- Provide a quick overview of **theory** and **CG applications**
  - Many extra slides in the course notes + example code
- Summarize progress in the last 3-5 years
  - We have attempted to organize them
  - Discuss the main **challenges and opportunities** specific to CG

Help Us Improve

Help Us Improve

- Our aim is to convey what we found to be relevant so far
- You are invited/encouraged to give feedback
  - Speakup. Please send us your criticism/comments/suggestions

Help Us Improve

- Our aim is to convey what we found to be relevant so far
- You are invited/encouraged to give feedback
  - Speakup. Please send us your criticism/comments/suggestions
  - Ask questions, please!

Help Us Improve

- Our aim is to convey what we found to be relevant so far
- You are invited/encouraged to give feedback
  - Speakup. Please send us your criticism/comments/suggestions
  - Ask questions, please!
- **Thanks to the many who helped so far with slides/comments**

Representations in Computer Graphics

Representations in Computer Graphics

- Images (e.g., pixel grid)
- Volume (e.g., voxel grid)
- Meshes (e.g., vertices/edges/faces)

Representations in Computer Graphics

- Images (e.g., pixel grid)
- Volume (e.g., voxel grid)
- Meshes (e.g., vertices/edges/faces)
- Point clouds (e.g., collection of points)

Representations in Computer Graphics

- Images (e.g., pixel grid)
- Volume (e.g., voxel grid)
- Meshes (e.g., vertices/edges/faces)
- Point clouds (e.g., collection of points)
- Animation (e.g., skeletal positions over time; cloth dynamics over time)

Representations in Computer Graphics

- Images (e.g., pixel grid)
- Volume (e.g., voxel grid)
- Meshes (e.g., vertices/edges/faces)
- Point clouds (e.g., collection of points)
- Animation (e.g., skeletal positions over time; cloth dynamics over time)
- Physics simulations (e.g., fluid flow over space-time, multi body interaction)

Problems in Computer Graphics

- Feature detection (image features, point features) $\mathbb{R}^{m\times m} \rightarrow \mathbb{Z}$
- Denoising, Smoothing, etc. $\mathbb{R}^{m\times m} \rightarrow \mathbb{R}^{m\times m}$
- Embedding, Metric learning $\mathbb{R}^{m\times m, m\times m} \rightarrow \mathbb{R}^{d}$
- Rendering $\mathbb{R}^{m\times m} \rightarrow \mathbb{R}^{m\times m}$
- Animation $\mathbb{R}^{3m\times t} \rightarrow \mathbb{R}^{3m}$
- Physical simulation $\mathbb{R}^{3m\times t} \rightarrow \mathbb{R}^{3m}$
- Generative models $\mathbb{R}^{d} \rightarrow \mathbb{R}^{m\times m}$

Problems in Computer Graphics

- Feature detection (image features, point features) $\mathbb{R}^{m \times m} \to \mathbb{Z}$
- Denoising, Smoothing, etc. $\mathbb{R}^{m \times m} \to \mathbb{R}^{m \times m}$
- Embedding, Metric learning $\mathbb{R}^{m \times m, m \times m} \to \mathbb{R}^d$

analysis

- Rendering $\mathbb{R}^{m \times m} \to \mathbb{R}^{m \times m}$
- Animation $\mathbb{R}^{3m \times t} \to \mathbb{R}^{3m}$
- Physical simulation $\mathbb{R}^{3m \times t} \to \mathbb{R}^{3m}$
- Generative models $\mathbb{R}^d \to \mathbb{R}^{m \times m}$

Problems in Computer Graphics

- Feature detection (image features, point features) $\mathbb{R}^{m\times m} \rightarrow \mathbb{Z}$
- Denoising, Smoothing, etc. $\mathbb{R}^{m\times m} \rightarrow \mathbb{R}^{m\times m}$
- Embedding, Metric learning $\mathbb{R}^{m\times m, m\times m} \rightarrow \mathbb{R}^{d}$

analysis

- Rendering $\mathbb{R}^{m\times m} \rightarrow \mathbb{R}^{m\times m}$
- Animation $\mathbb{R}^{3m\times t} \rightarrow \mathbb{R}^{3m}$
- Physical simulation $\mathbb{R}^{3m\times t} \rightarrow \mathbb{R}^{3m}$
- Generative models $\mathbb{R}^{d} \rightarrow \mathbb{R}^{m\times m}$

synthesis

Goal: Learn a Parametric Function

$$f_\theta : \mathbb{X} \longrightarrow \mathbb{Y}$$

θ: function parameters, these are learned

$\mathbb{X}$: source domain

$\mathbb{Y}$: target domain

Goal: Learn a Parametric Function

$$f_\theta : \mathbb{X} \longrightarrow \mathbb{Y}$$

θ: function parameters, these are learned

$\mathbb{X}$: source domain

$\mathbb{Y}$: target domain

Examples:

Goal: Learn a Parametric Function

$$f_\theta : \mathbb{X} \longrightarrow \mathbb{Y}$$

θ: function parameters, these are learned

$\mathbb{X}$: source domain

$\mathbb{Y}$: target domain

Examples:

Image Classification: $f_\theta : \mathbb{R}^{w \times h \times c} \longrightarrow \{0, 1, \ldots, k-1\}$

$w \times h \times c$: image dimensions

k: class count

Goal: Learn a Parametric Function

$$f_\theta : \mathbb{X} \longrightarrow \mathbb{Y}$$

θ: function parameters, these are learned

$\mathbb{X}$: source domain

$\mathbb{Y}$: target domain

Examples:

Image Classification: $f_\theta : \mathbb{R}^{w \times h \times c} \longrightarrow \{0, 1, \ldots, k-1\}$

$w \times h \times c$: image dimensions

k: class count

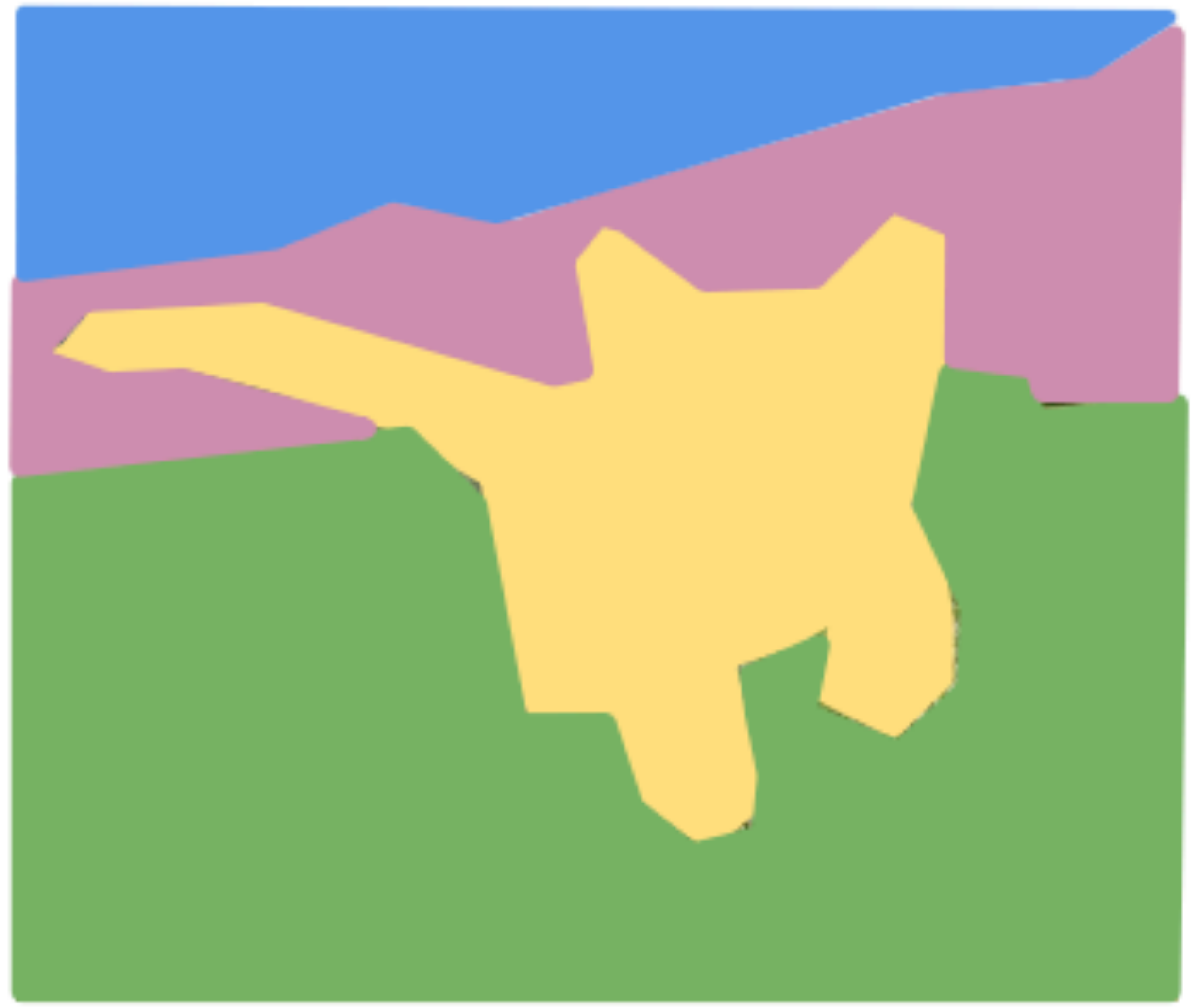
Image Synthesis: $f_\theta : \mathbb{R}^n \longrightarrow \mathbb{R}^{w \times h \times c}$

n : latent variable count

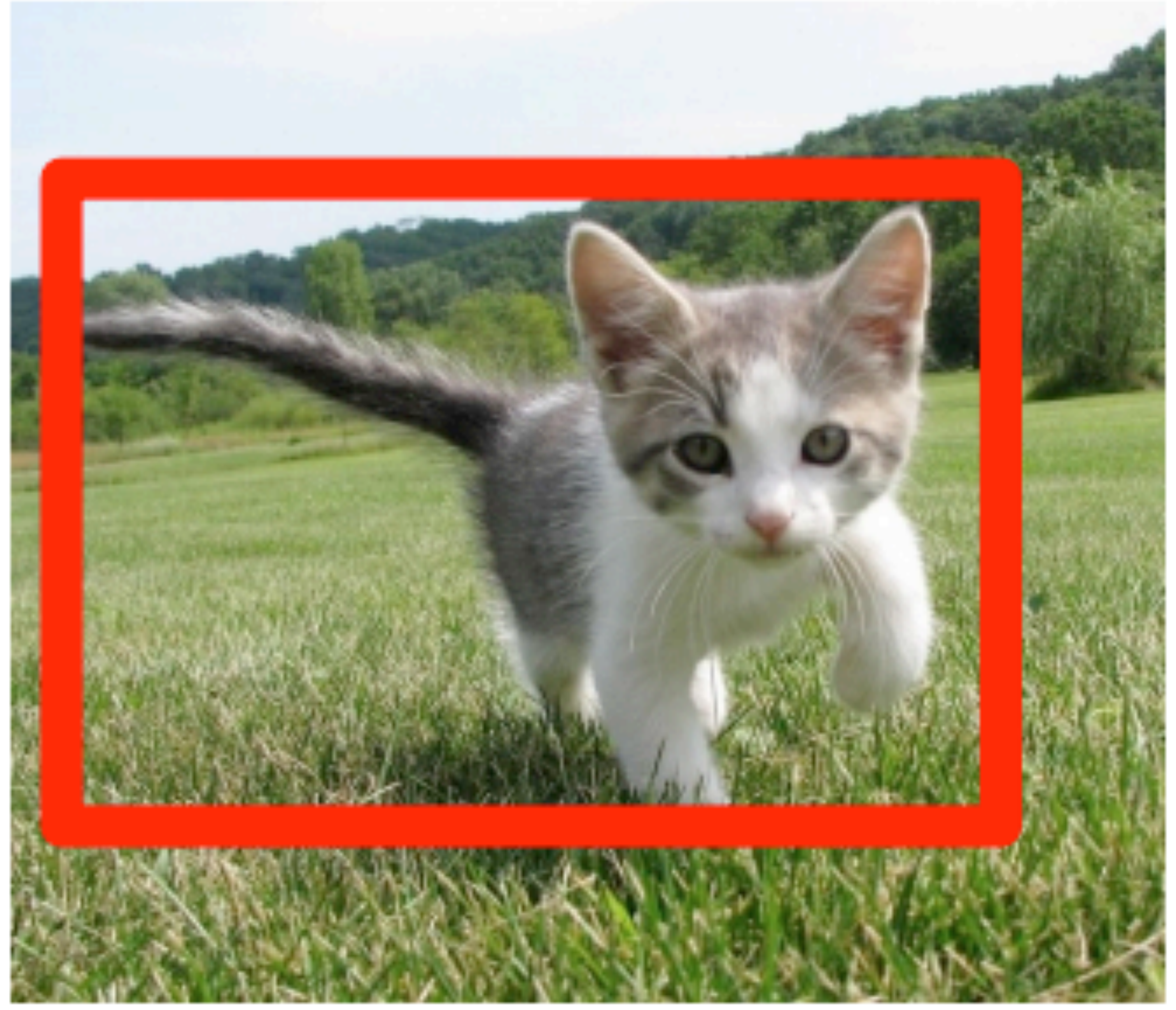
$w \times h \times c$: image dimensions

Semantic Segmentation

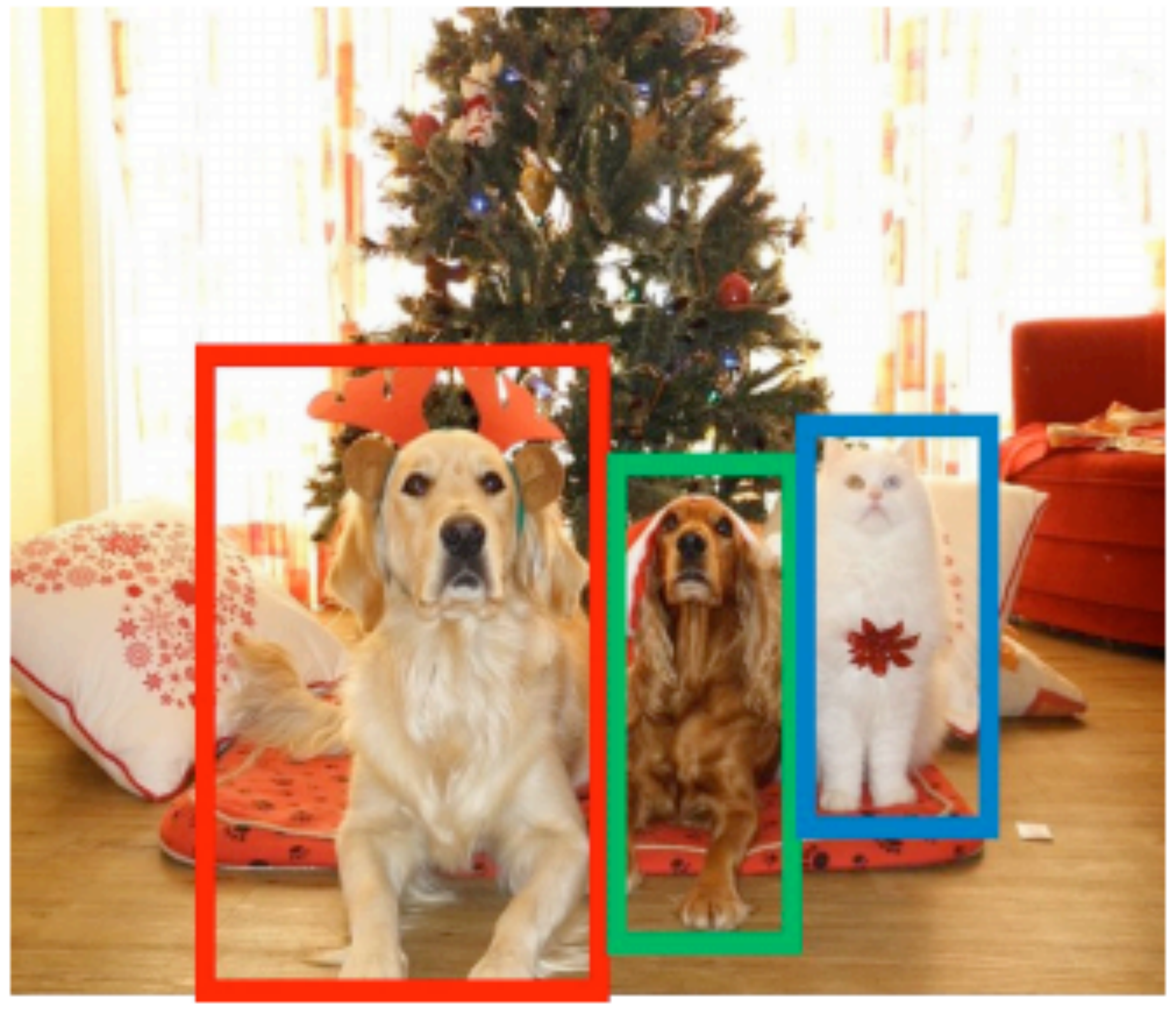
Semantic Segmentation



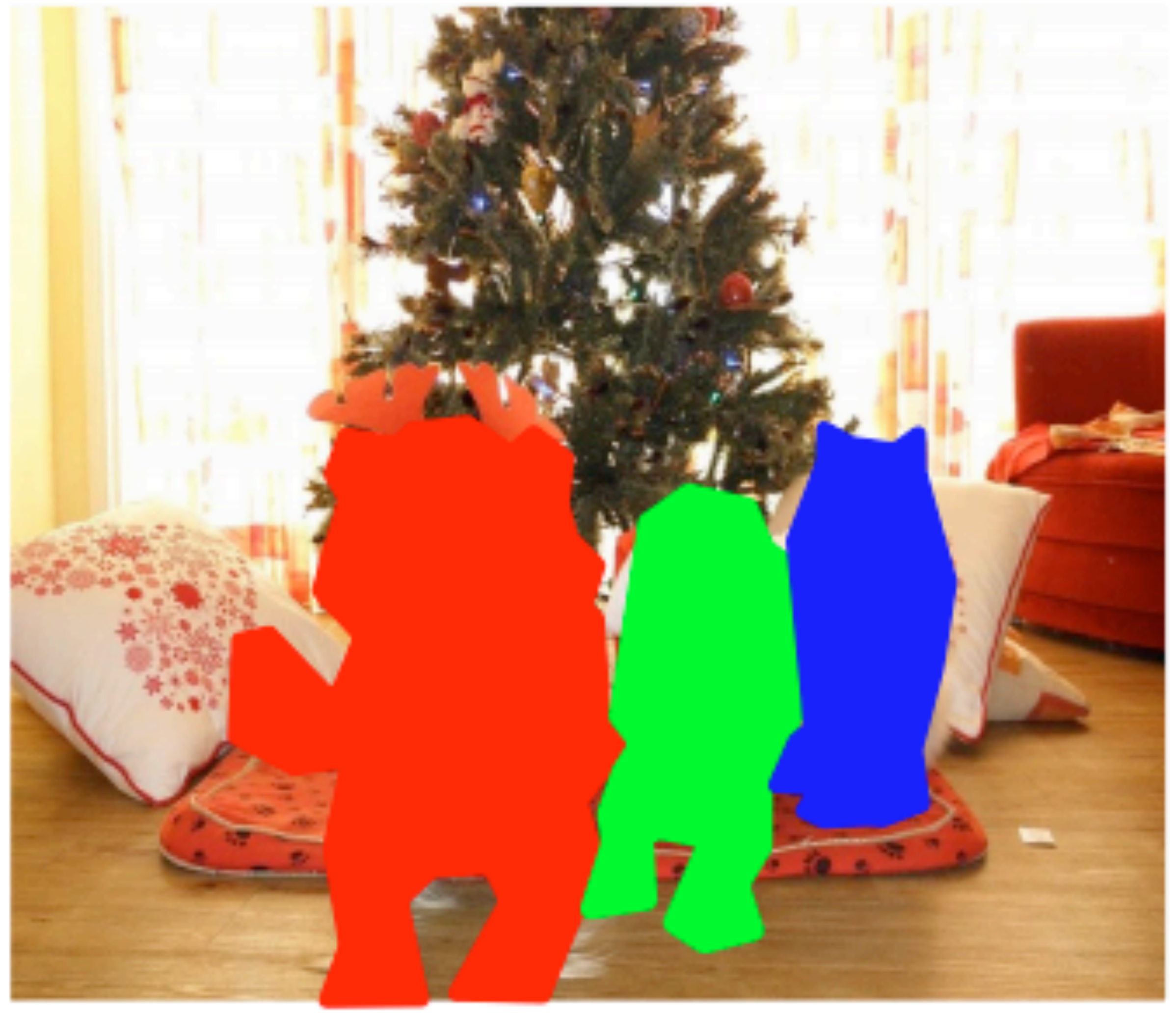
Classification + Localization



Object Detection



Instance Segmentation



http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf

The Legend of Tarzan

Pose Detection using CNNs

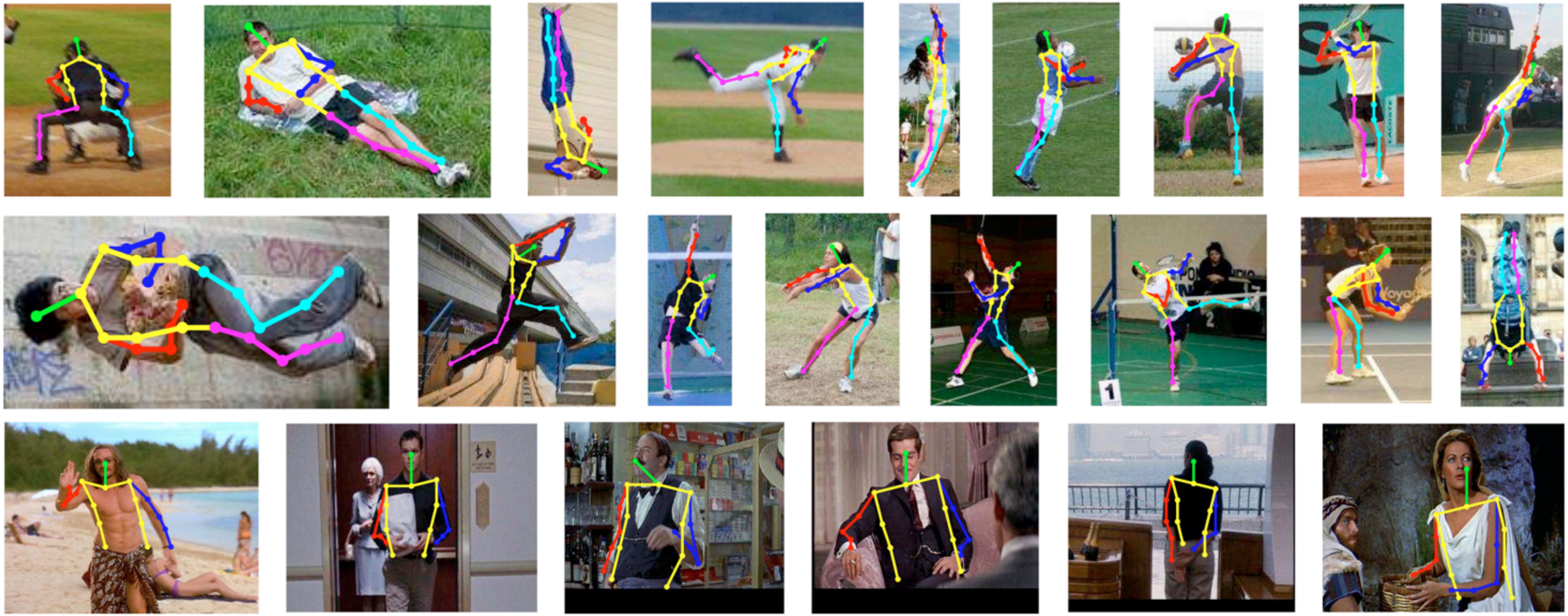


Image Denoising

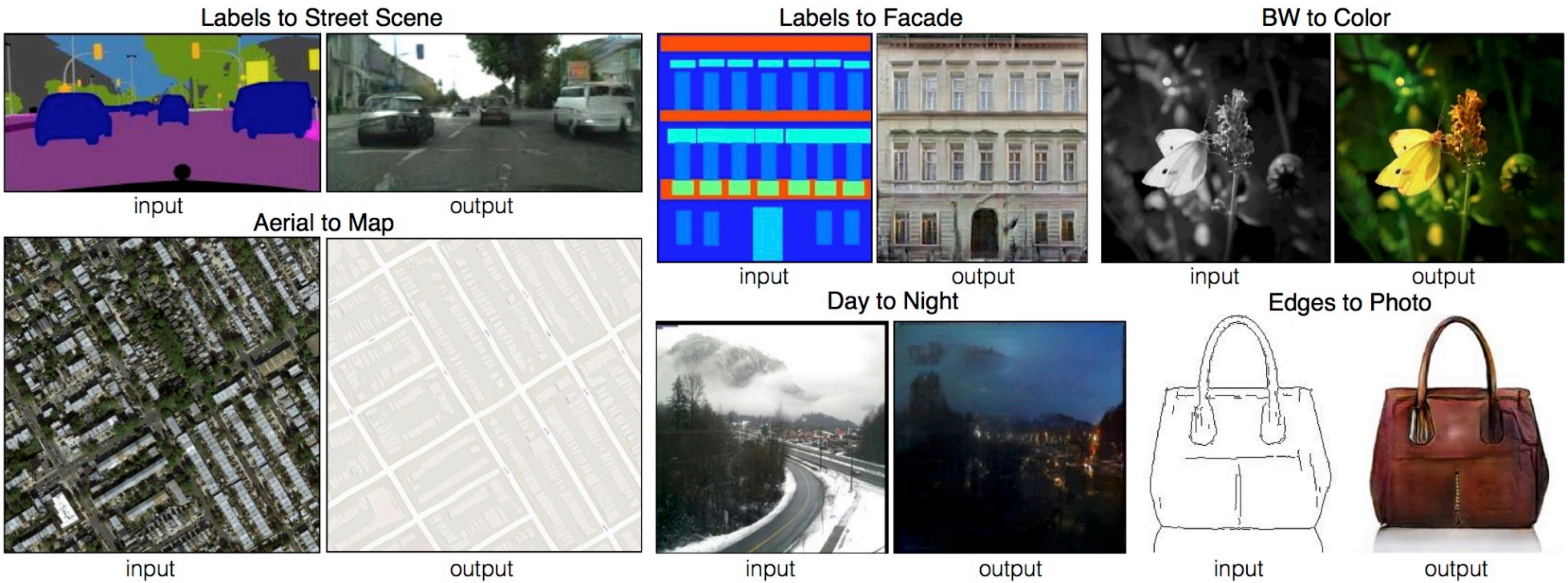
[Chaitanya et al. 2017, Siggraph]

Image Denoising

[Chaitanya et al. 2017, Siggraph]

Image Translation Problems

[Isola et al. 2017, CVPR]



Sketch to Face!

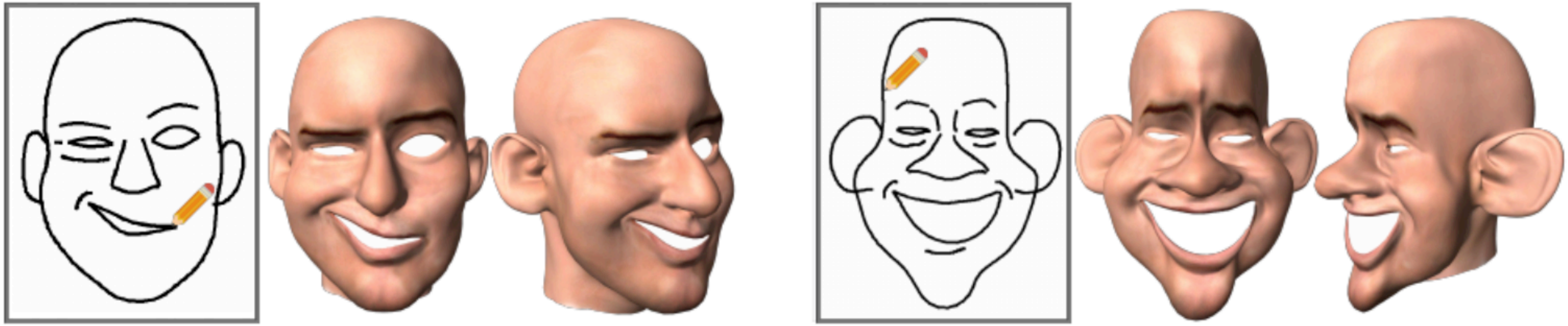
[Han et al. 2017, Siggraph]

DeepSketch2Face: A Deep Learning Based Sketching System for 3D Face and Caricature Modeling

Sketch to Face!

[Han et al. 2017, Siggraph]

DeepSketch2Face: A Deep Learning Based Sketching System for 3D Face and Caricature Modeling



[Wang et al. 2018, Siggraph Asia]

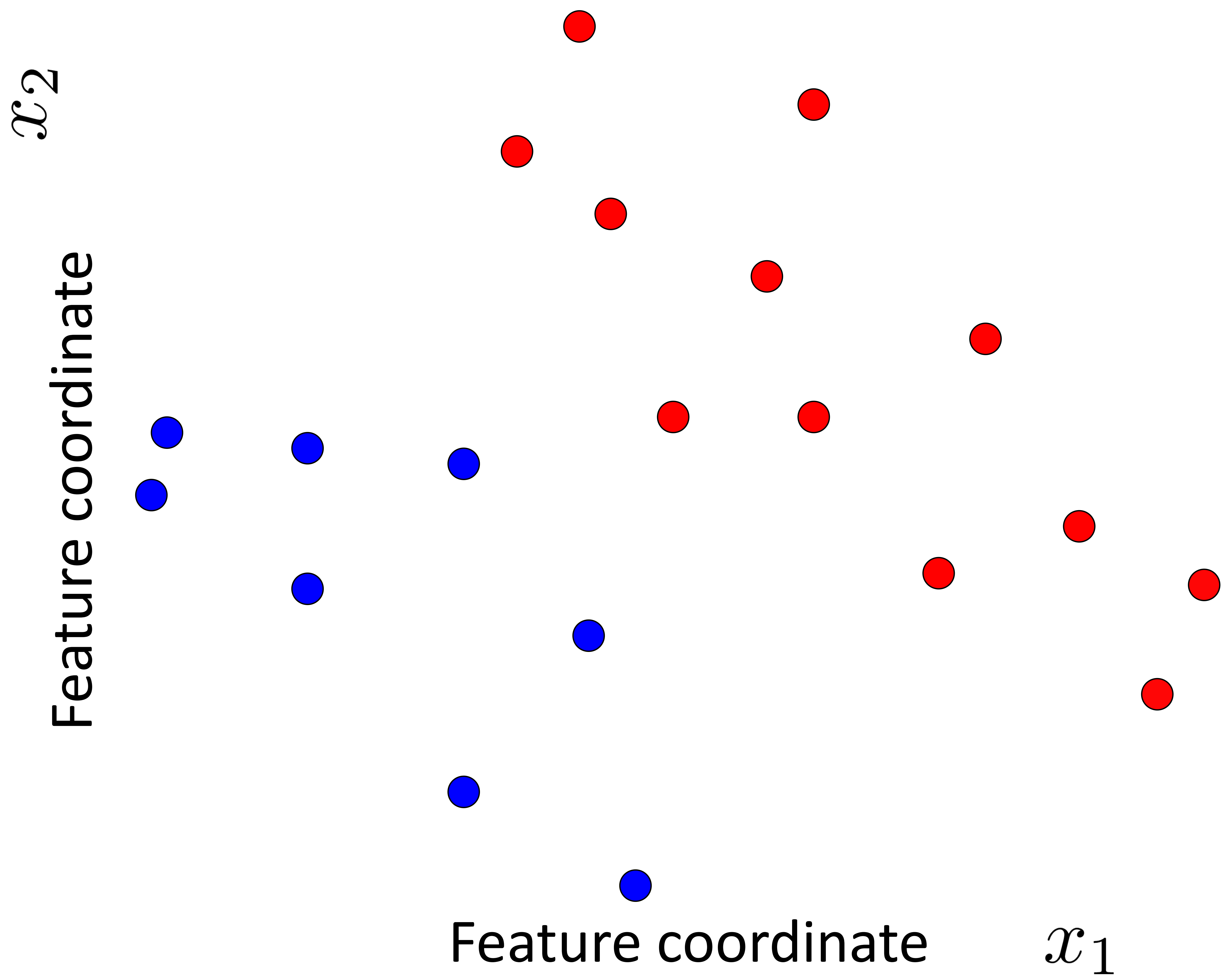
Real Images

[Wang et al. 2018, Siggraph Asia]

Real Images

Machine Learning 101: Linear Classifier

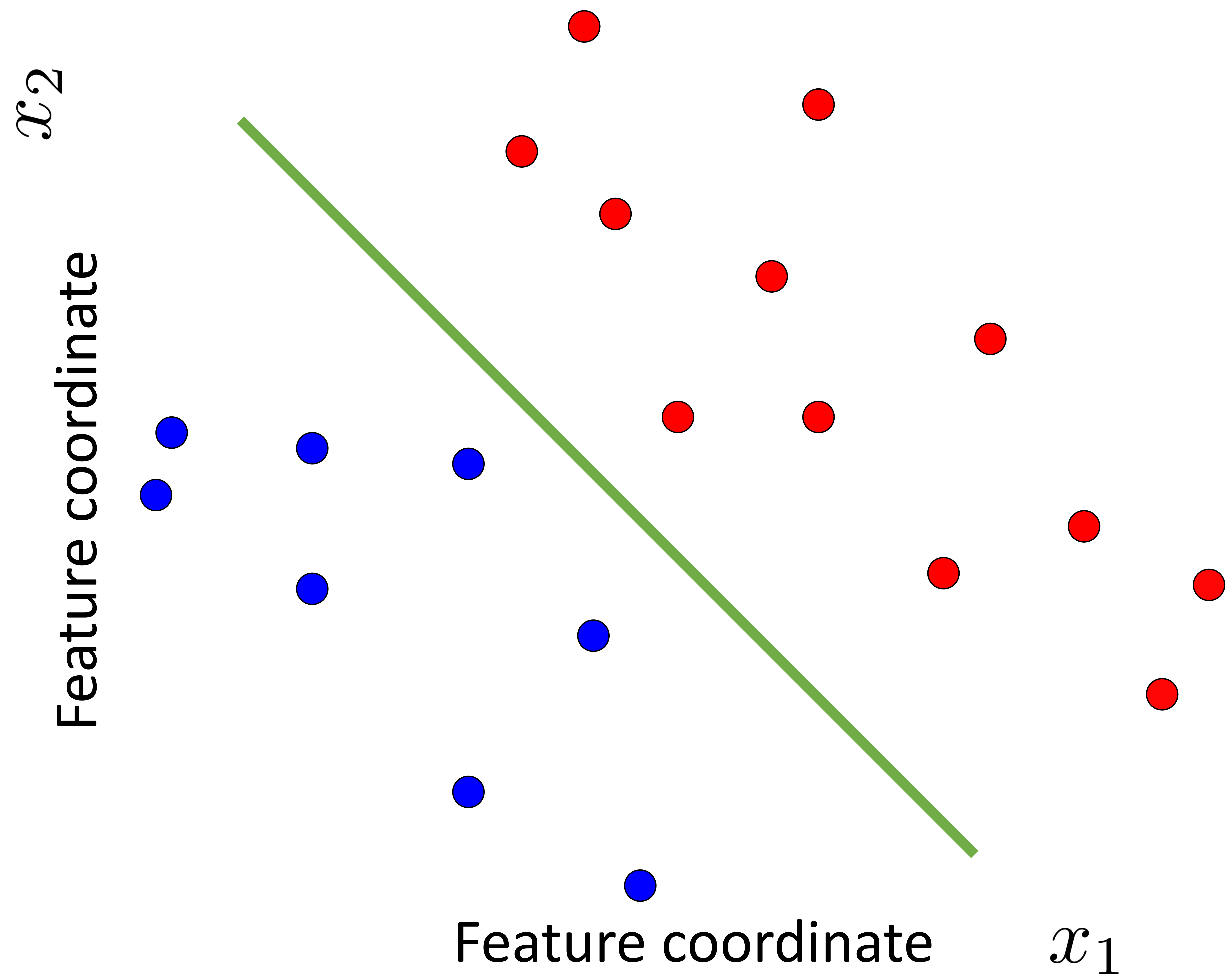
$$f_\theta : \mathbb{R}^n \longrightarrow \{0, 1\}$$



Each data point has a class label:

$$y^i = \begin{cases} 1 & (\textcolor{red}{\bullet}) \\ 0 & (\textcolor{blue}{\bullet}) \end{cases}$$

Machine Learning 101: Linear Classifier

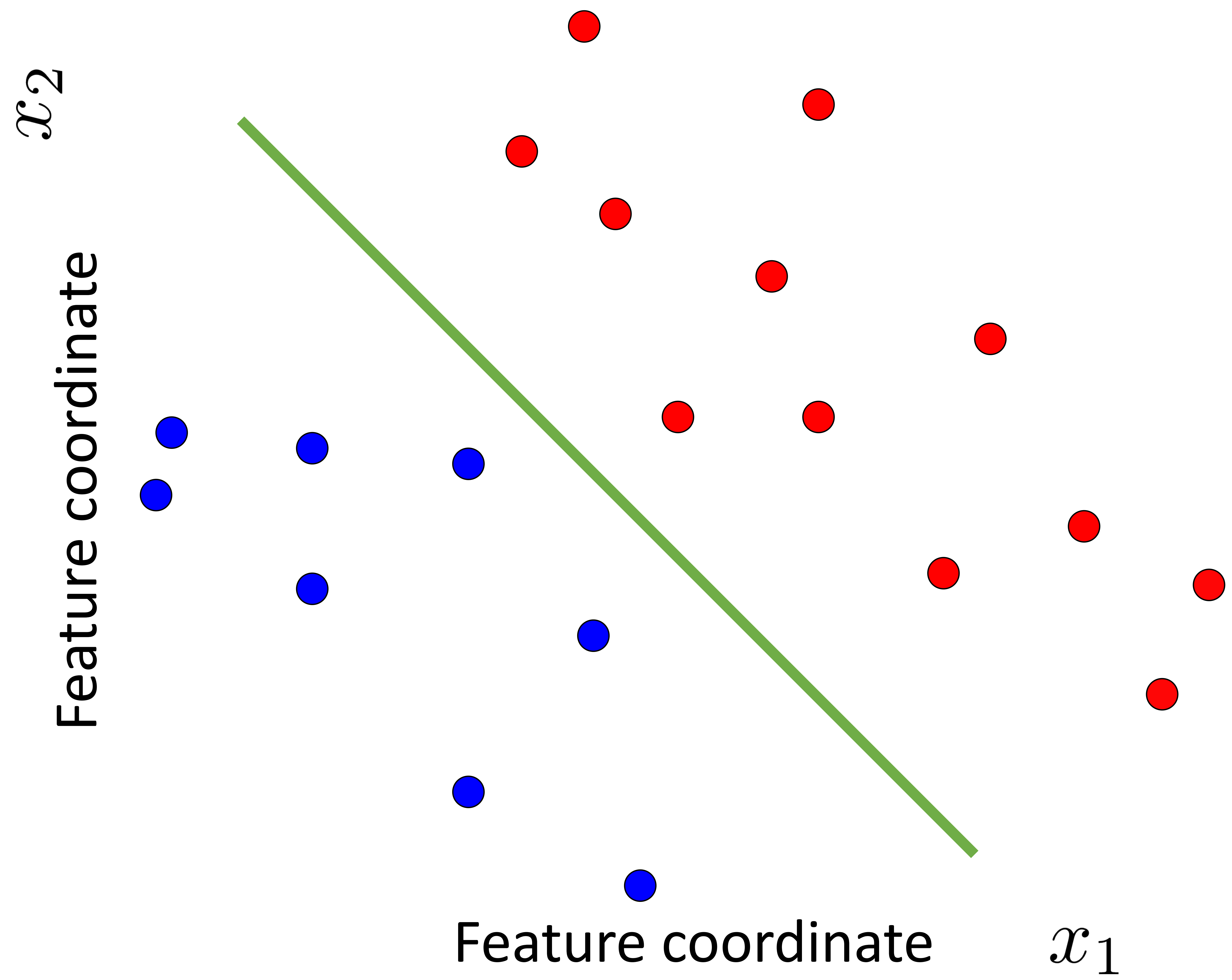


$$f_\theta : \mathbb{R}^n \longrightarrow \{0, 1\}$$

Each data point has a class label:

$$y^i = \begin{cases} 1 & (\textcolor{red}{\bullet}) \\ 0 & (\textcolor{blue}{\bullet}) \end{cases}$$

Machine Learning 101: Linear Classifier



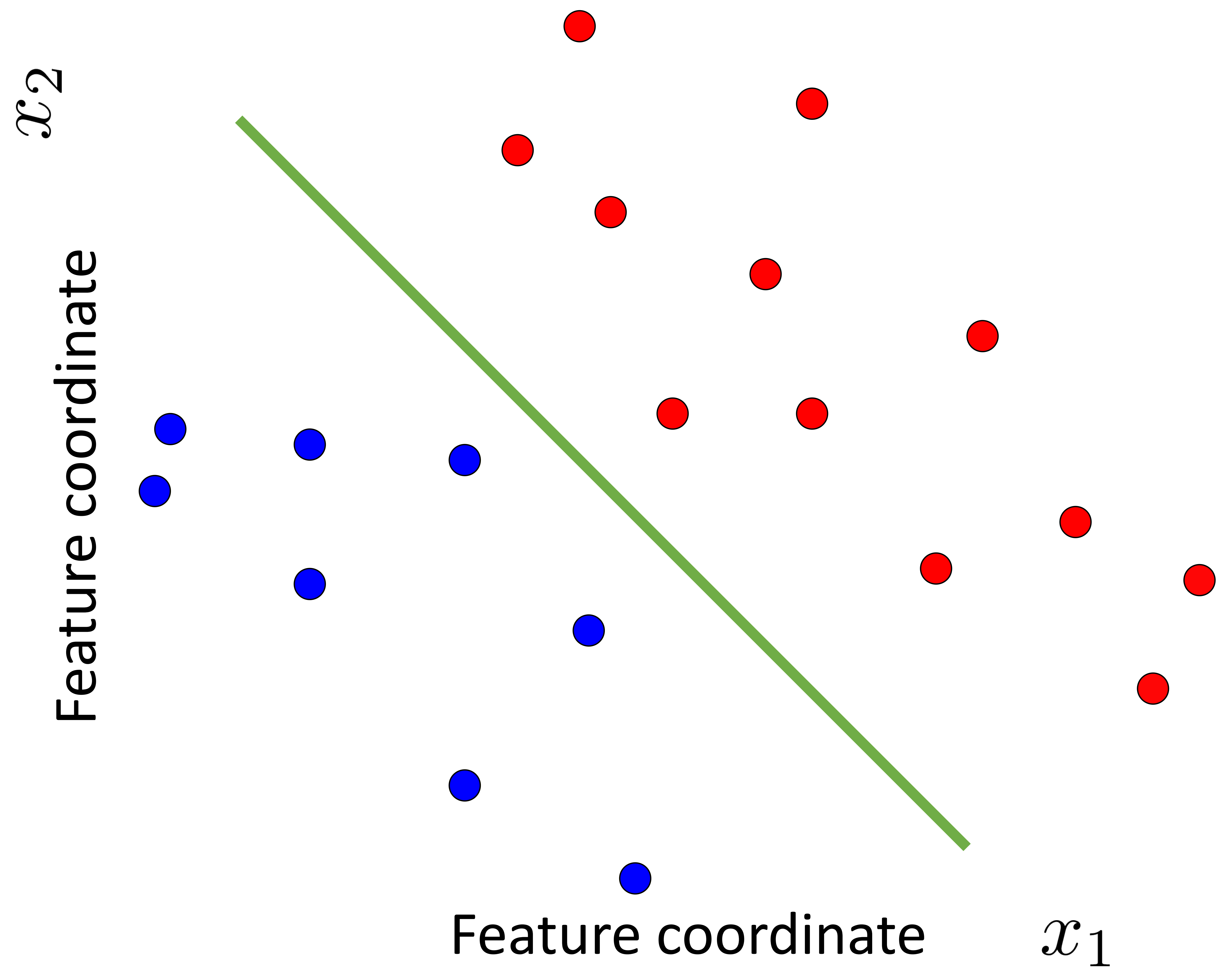
$$f_\theta : \mathbb{R}^n \longrightarrow \{0, 1\}$$

$$f_\theta(x) = \begin{cases} 1 & \text{if } wx + b \geq 0 \\ 0 & \text{if } wx + b < 0 \end{cases}$$

Each data point has a class label:

$$y^i = \begin{cases} 1 & (\bullet) \\ 0 & (\bullet) \end{cases}$$

Machine Learning 101: Linear Classifier



$$f_\theta : \mathbb{R}^n \longrightarrow \{0, 1\}$$

$$f_\theta(x) = \begin{cases} 1 & \text{if } wx + b \geq 0 \\ 0 & \text{if } wx + b < 0 \end{cases}$$

$$\boxed{\theta = \{w, b\}}$$

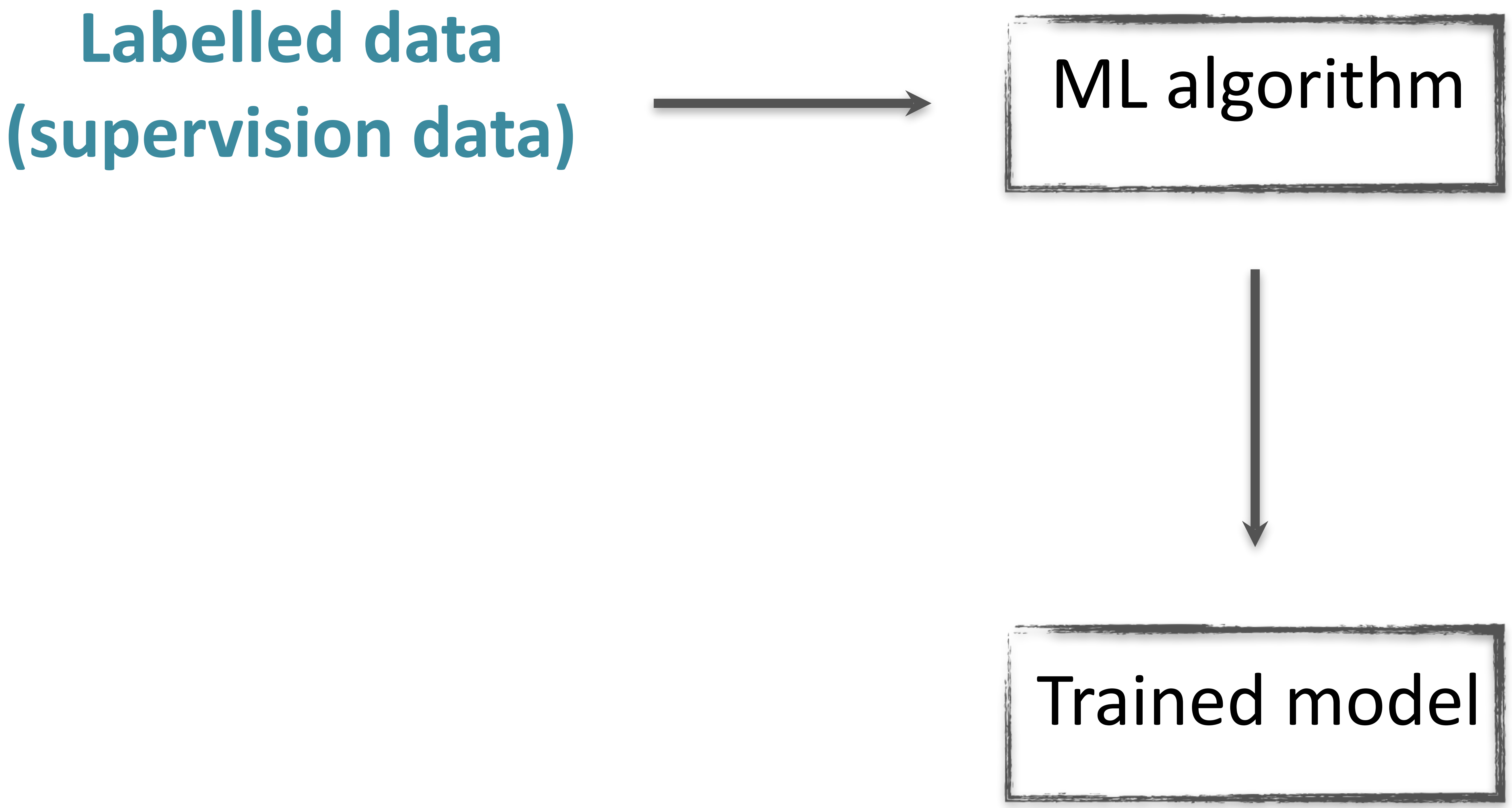
Each data point has a class label:

$$y^i = \begin{cases} 1 & (\ ) \\ 0 & (\ ) \end{cases}$$

Data-driven Algorithms (Supervised)

Labelled data
(supervision data)

Data-driven Algorithms (Supervised)



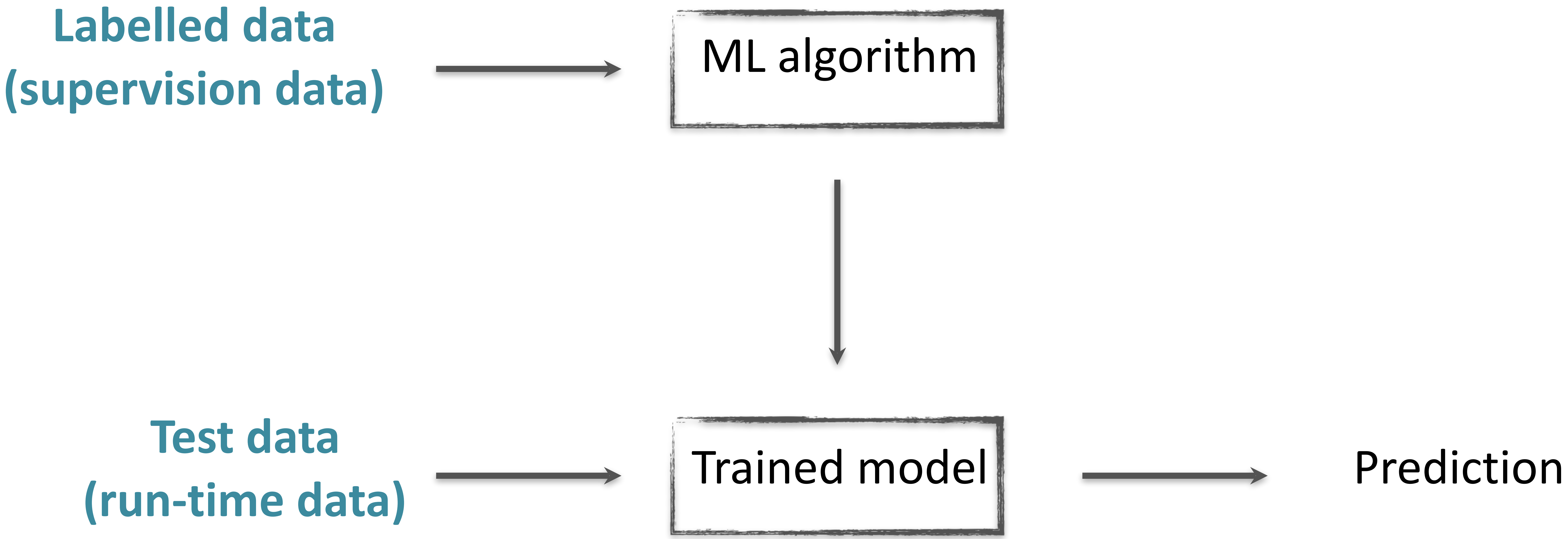
Data-driven Algorithms (Supervised)

Labelled data
(supervision data) → ML algorithm

ML algorithm ↓ Trained model

Test data
(run-time data) → Trained model

Data-driven Algorithms (Supervised)



Data-driven Algorithms (Supervised)

Labelled data
(supervision data)

ML algorithm

converged?

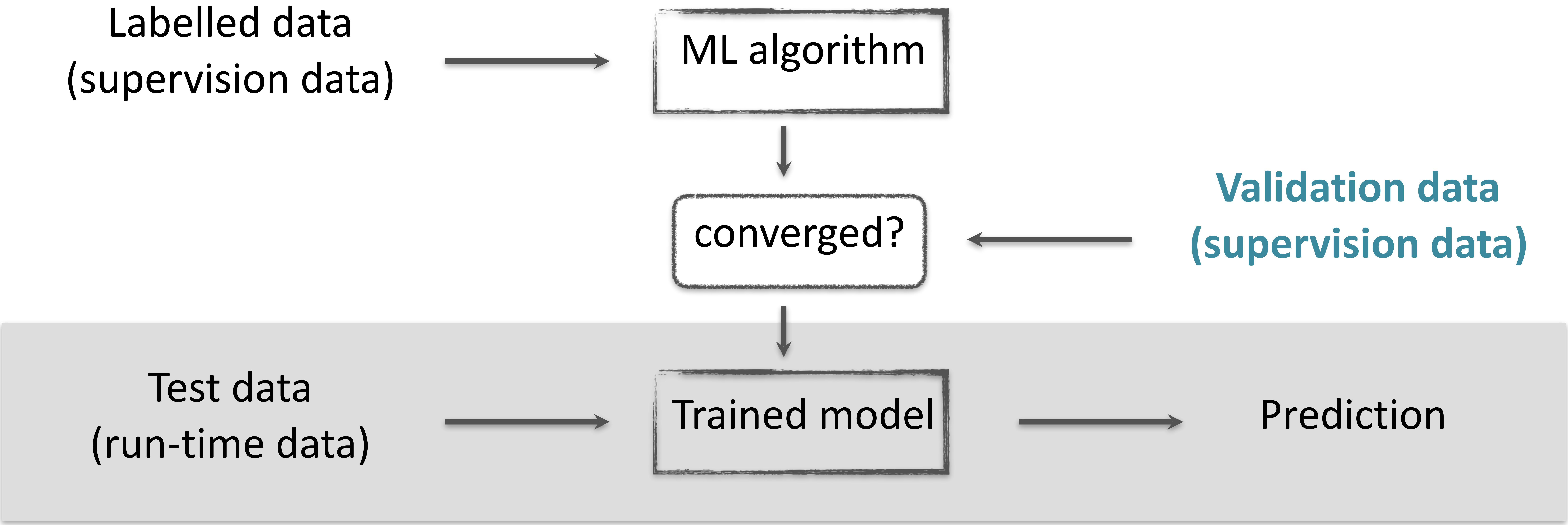
Validation data
(supervision data)

Test data
(run-time data)

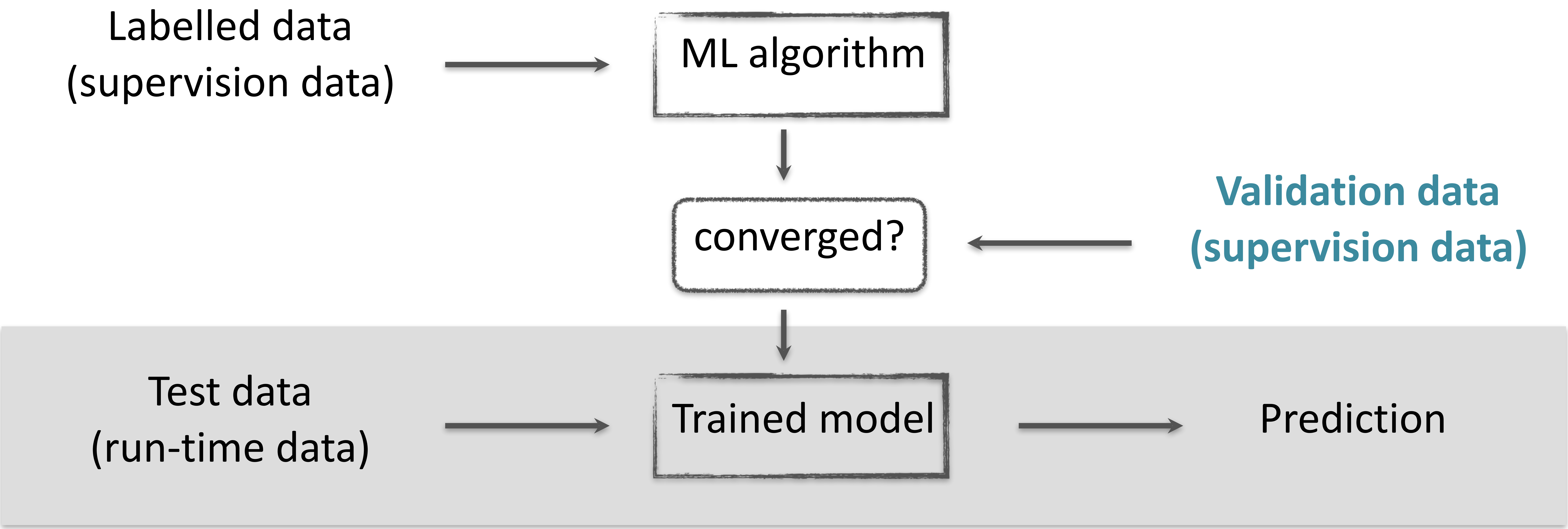
Trained model

Prediction

Data-driven Algorithms (Supervised)

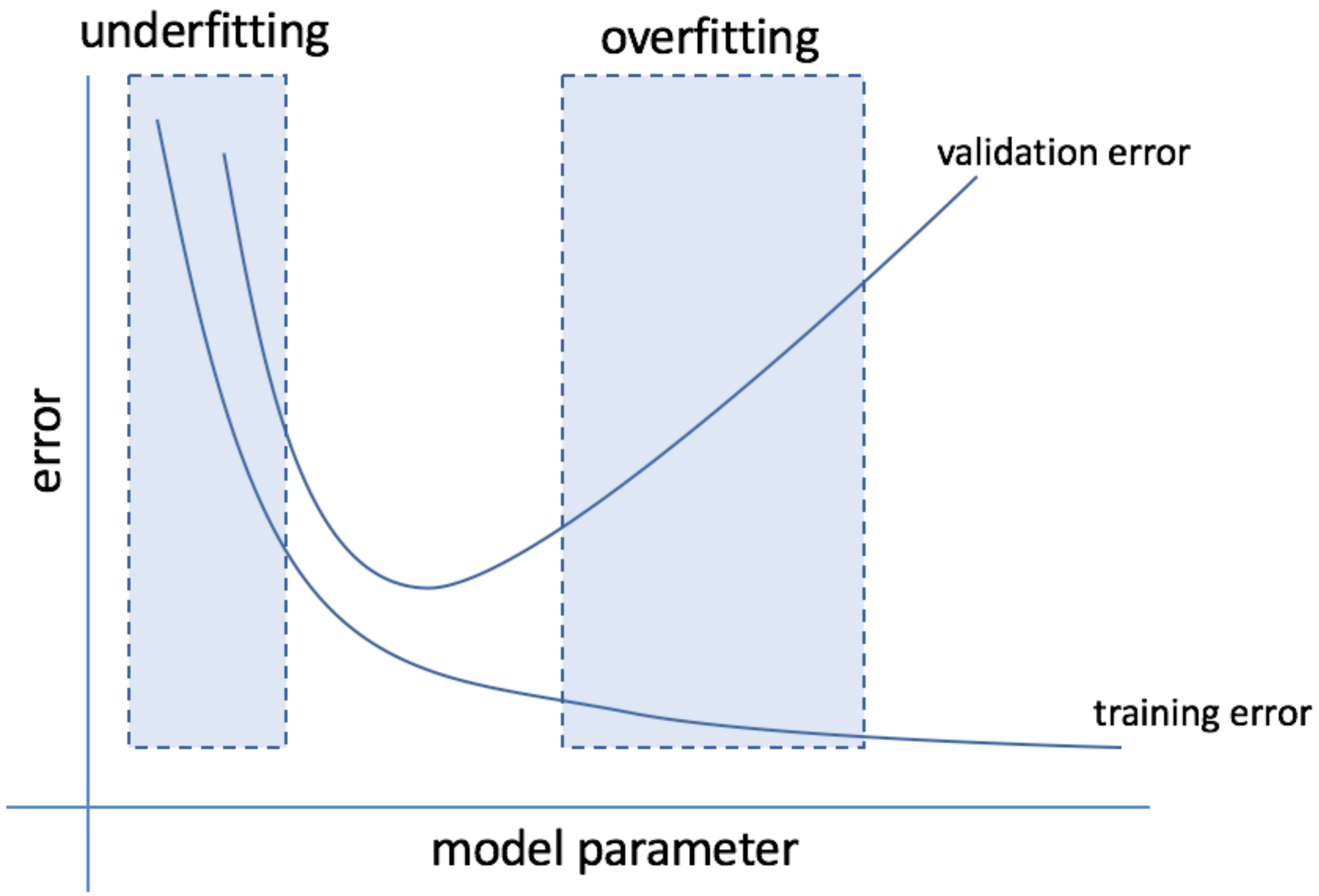


Data-driven Algorithms (Supervised)

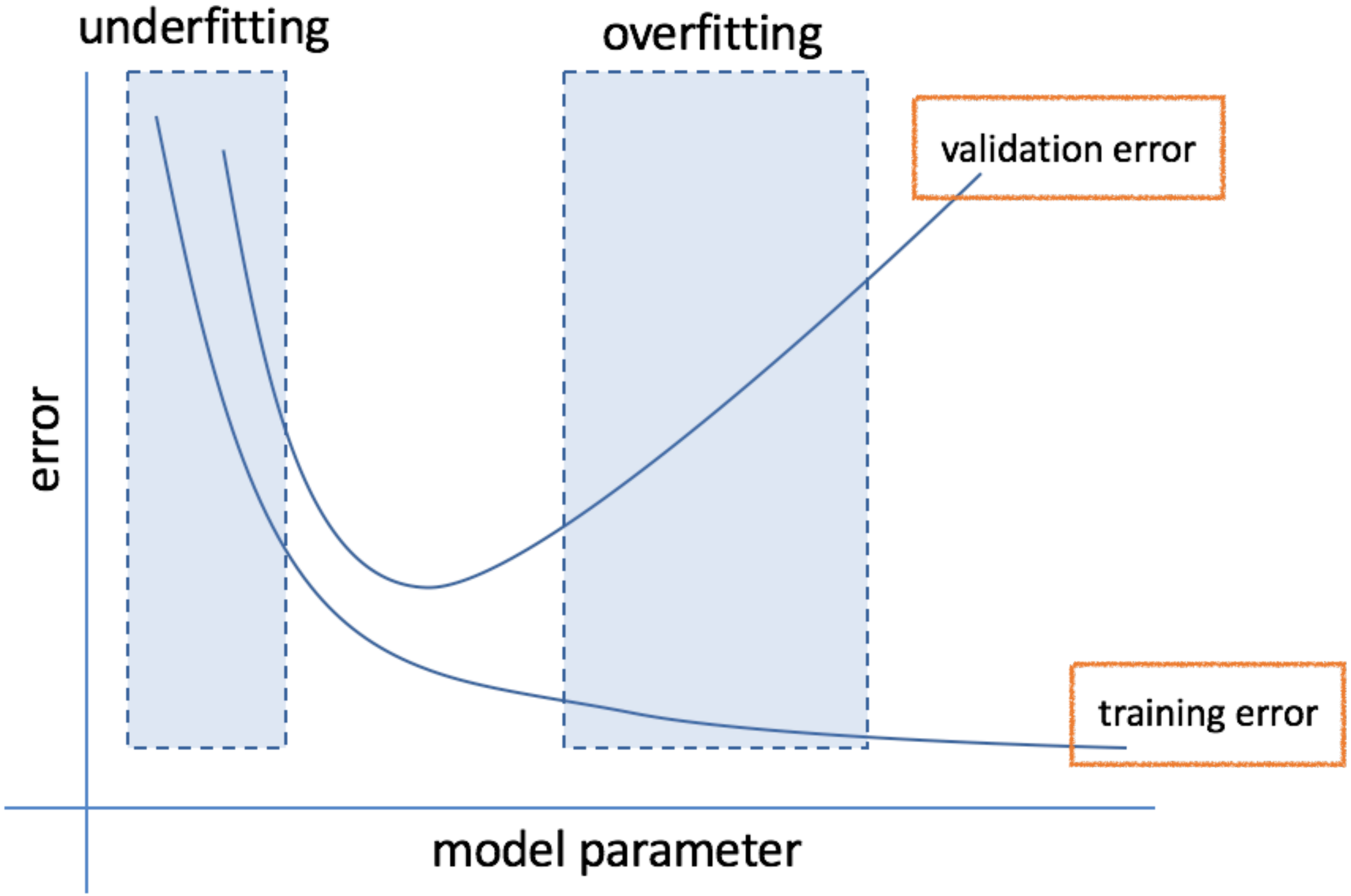


Implementation Practice: Training: 70%; Validation: 15%; Test 15%

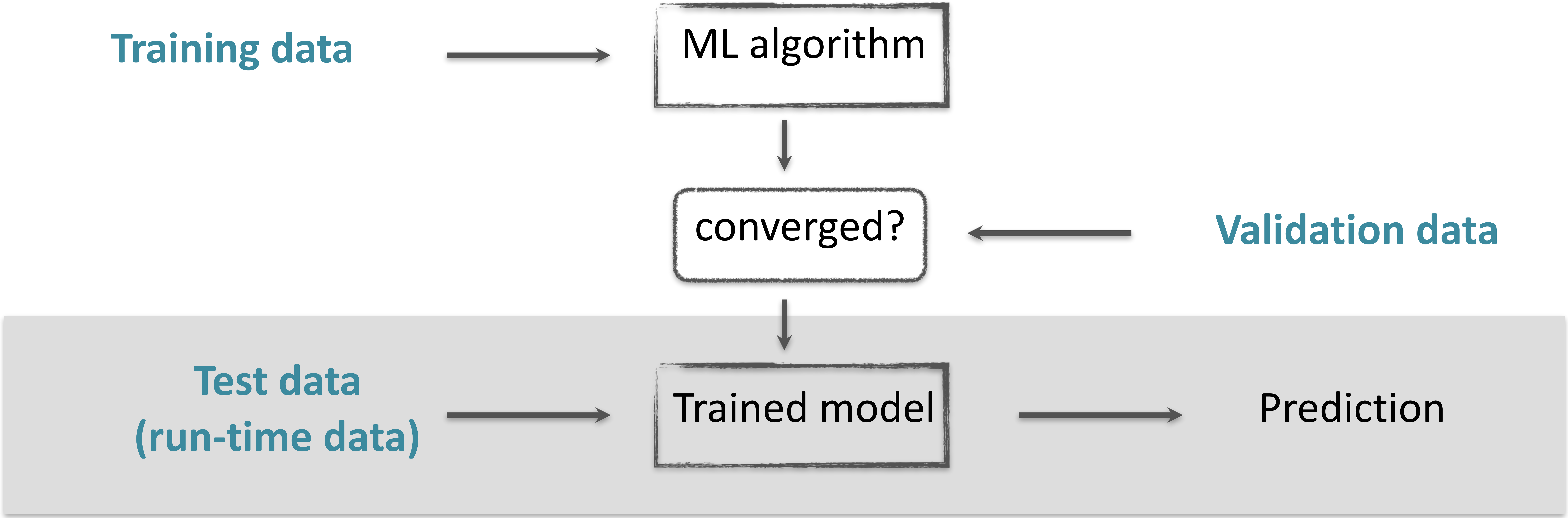
Training versus Validation Loss/Accuracy



Training versus Validation Loss/Accuracy

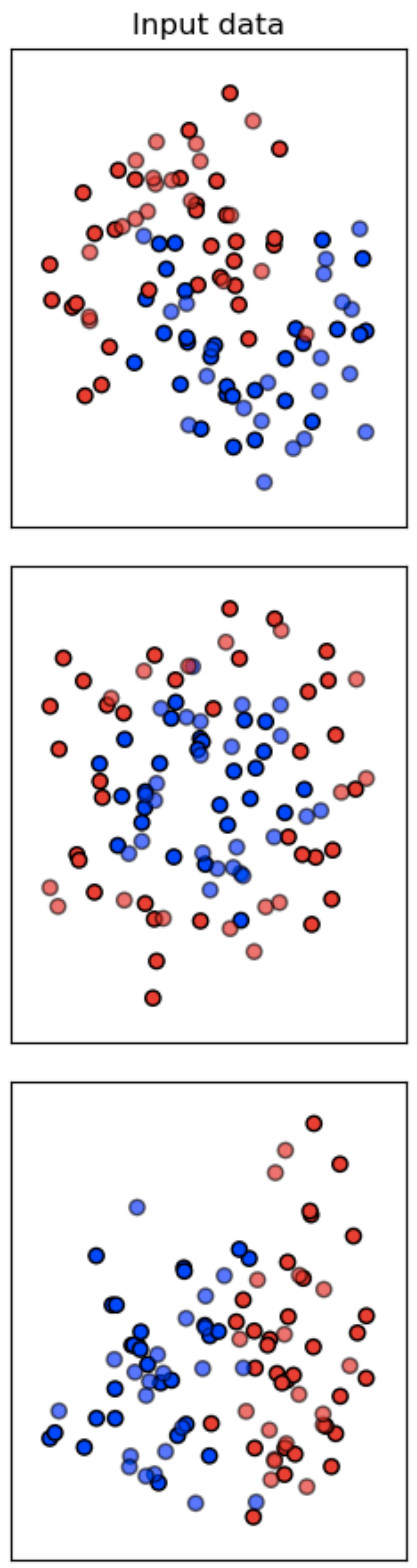


Data-driven Algorithms (Unsupervised)



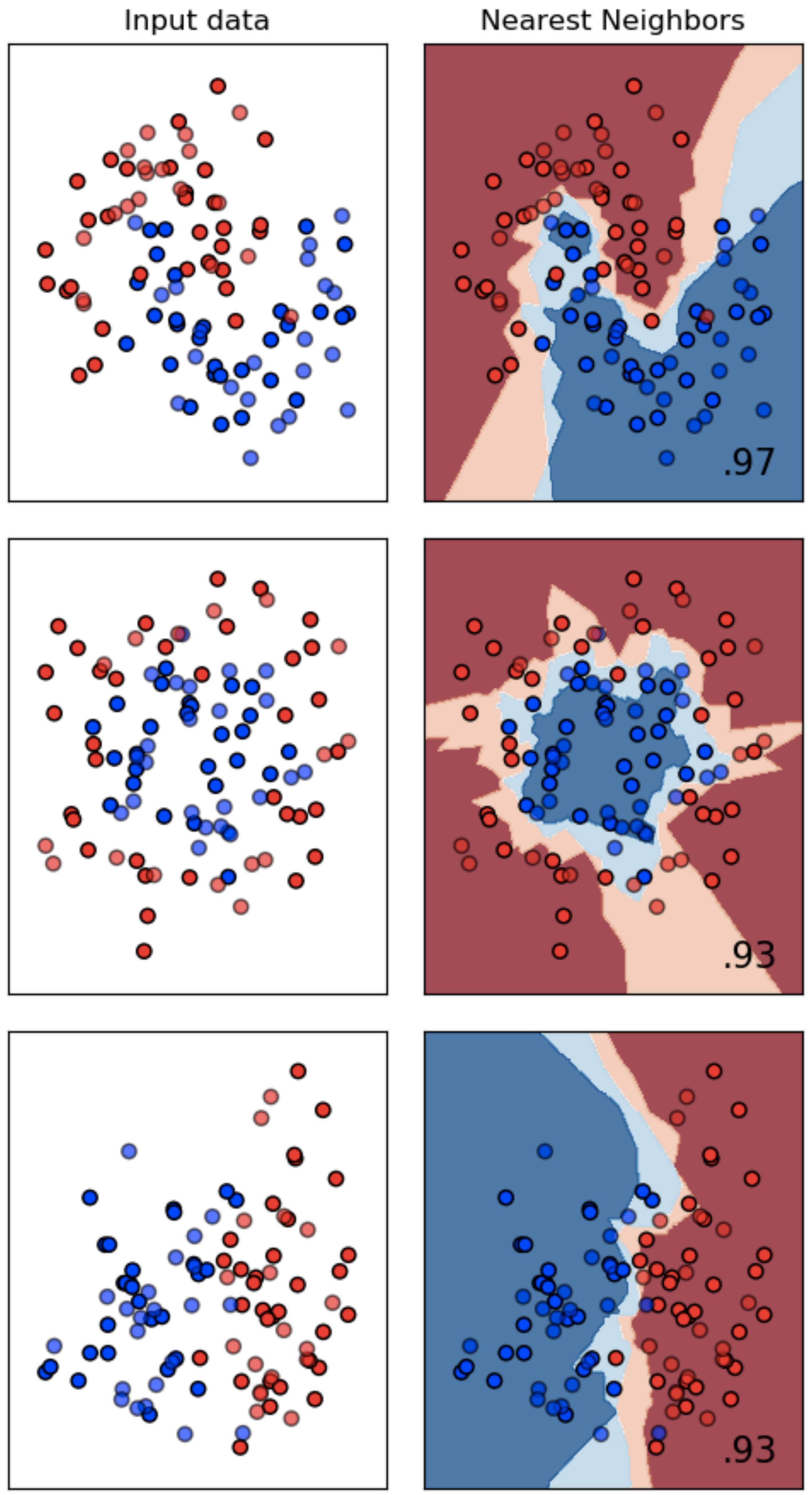
Implementation Practice: Training: 70%; Validation: 15%; Test 15%

Various ML Approaches (Supervised approaches)



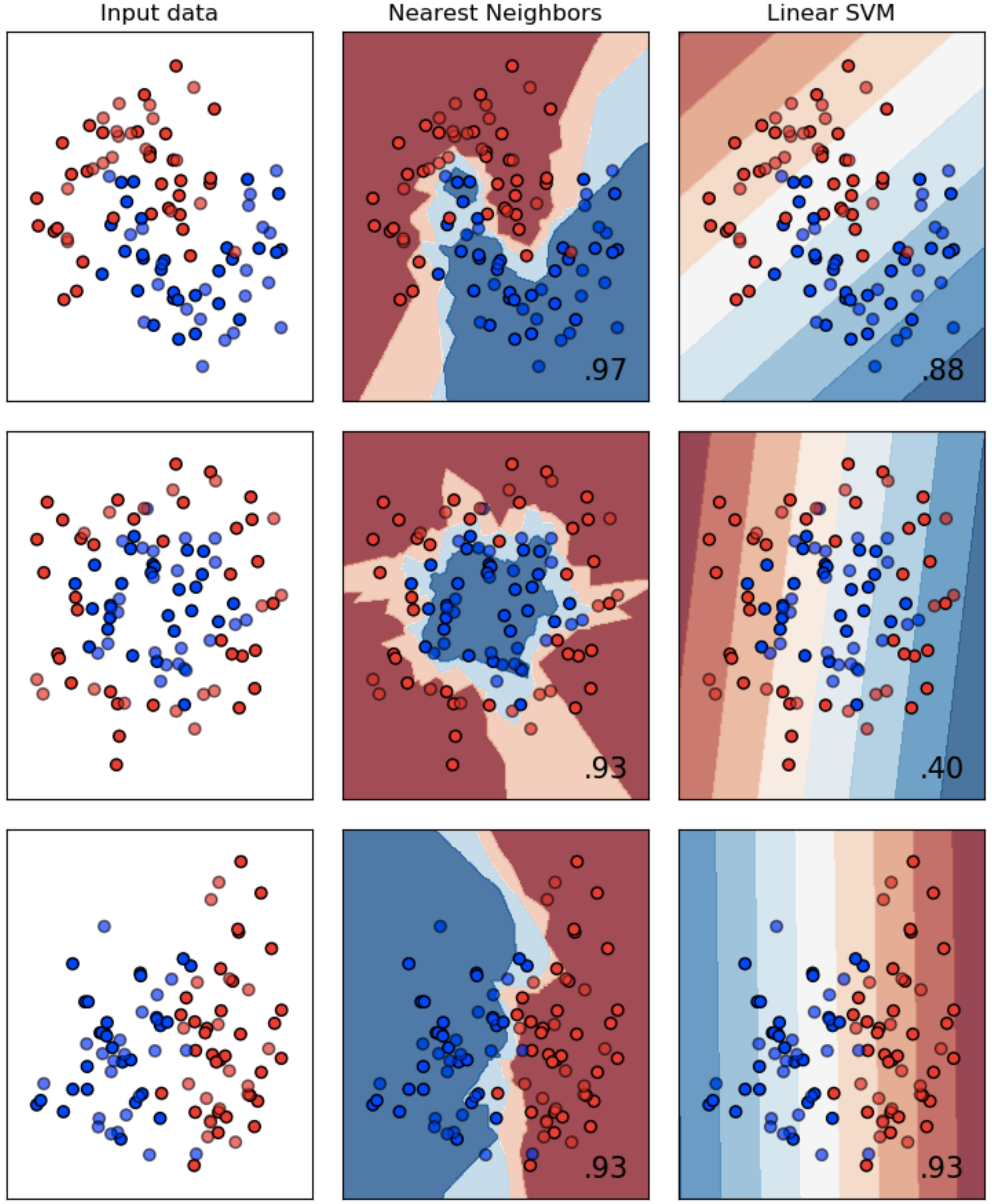
http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

Various ML Approaches (Supervised approaches)



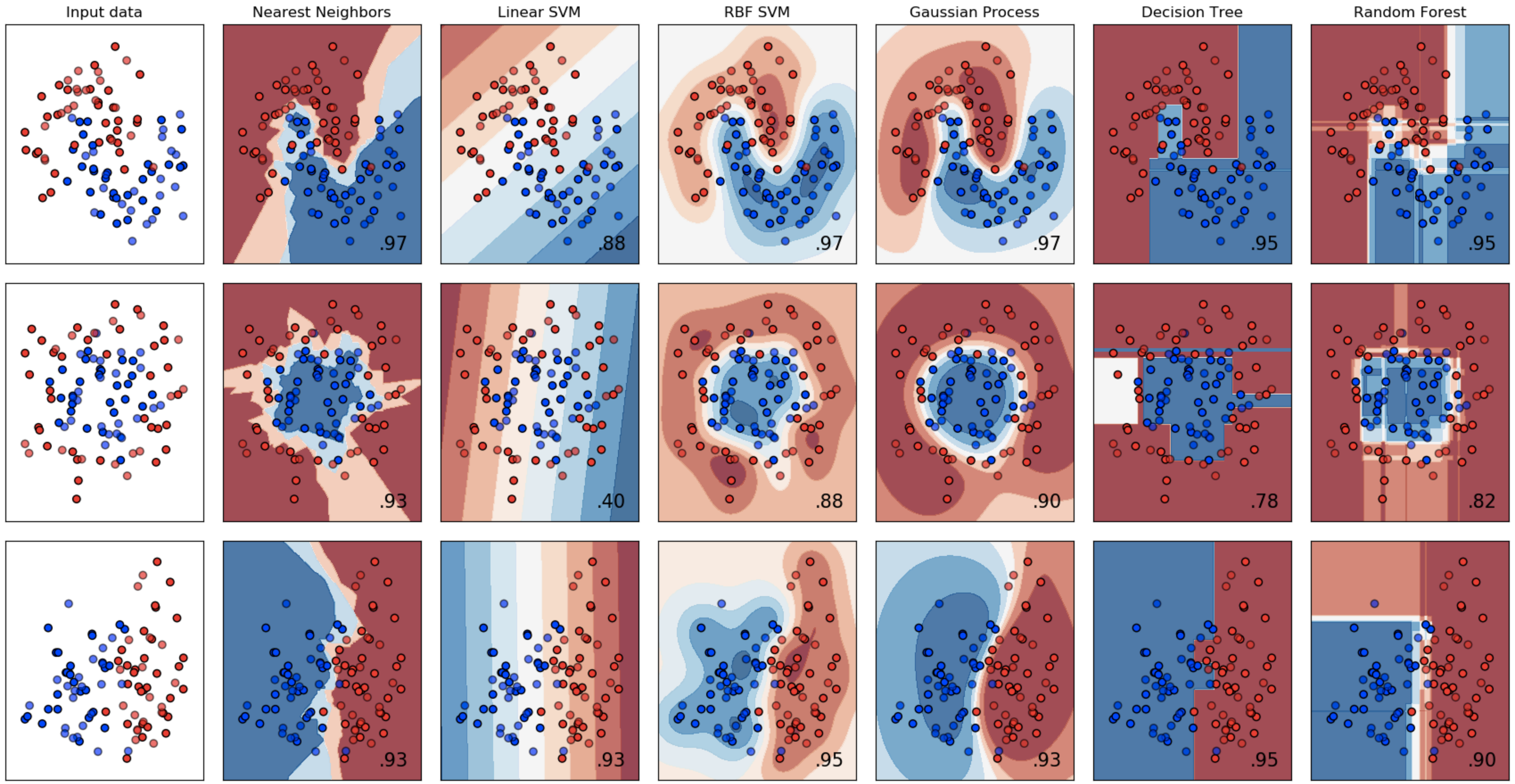
http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

Various ML Approaches (Supervised approaches)



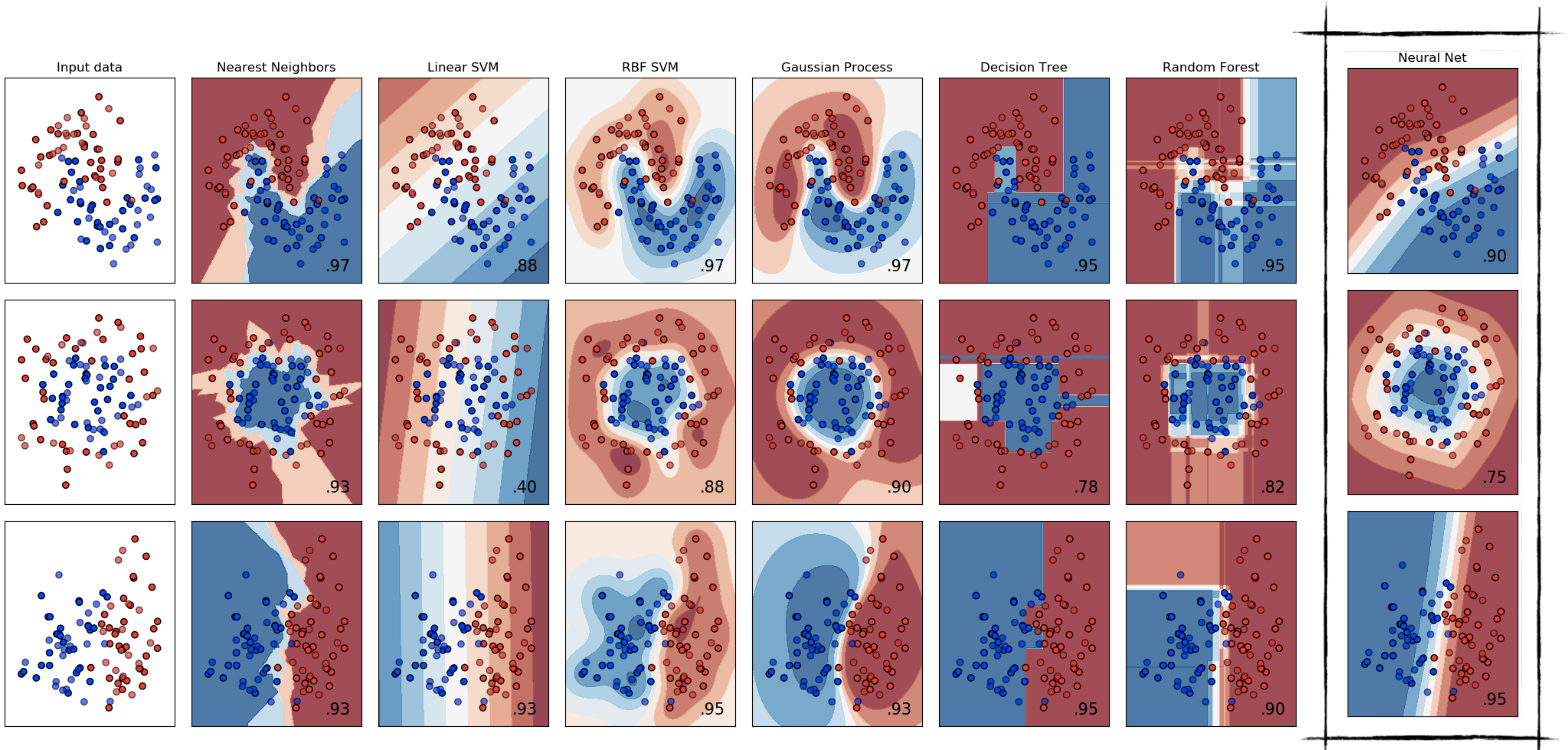
http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

Various ML Approaches (Supervised approaches)



http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

Various ML Approaches (Supervised approaches)



http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

Rise of Learning

- 1958: Perceptron
- 1974: Backpropagation
- 1981: Hubel & Wiesel wins Nobel prize for ‘visual system’
- 1990s: SVM era
- 1998: CNN used for handwriting analysis
- **2012: AlexNet wins ImageNet**

What is Special about CG?

What is Special about CG?

1. **Regular data structure** and easy to parallelize
   (e.g., image translation)

2. Many sources of input data — **model building**
   (e.g., images, scanners, motion capture)

What is Special about CG?

1. **Regular data structure** and easy to parallelize
   (e.g., image translation)

2. Many sources of input data — **model building**
   (e.g., images, scanners, motion capture)

3. Many sources of **synthetic data** — can serve as supervision data
   (e.g., rendering, animation)

What is Special about CG?

1. **Regular data structure** and easy to parallelize
   (e.g., image translation)

2. Many sources of input data — **model building**
   (e.g., images, scanners, motion capture)

3. Many sources of **synthetic data** — can serve as supervision data
   (e.g., rendering, animation)

4. Many problems in **generative models** and need for **user-control**

Main Challenges and Scope for Innovation

Main Challenges and Scope for Innovation

1. **Representation:** How is the data organised and structured?

2. **Training data:** Is it synthetic or real, or mixed?

Main Challenges and Scope for Innovation

1. **Representation:** How is the data organised and structured?

2. **Training data:** Is it synthetic or real, or mixed?

3. **User control:** End-to-end or in small steps?

Main Challenges and Scope for Innovation

1. **Representation:** How is the data organised and structured?

2. **Training data:** Is it synthetic or real, or mixed?

3. **User control:** End-to-end or in small steps?

4. **Loss functions:** Hand-crafted or learned from data?

Data is the New Currency

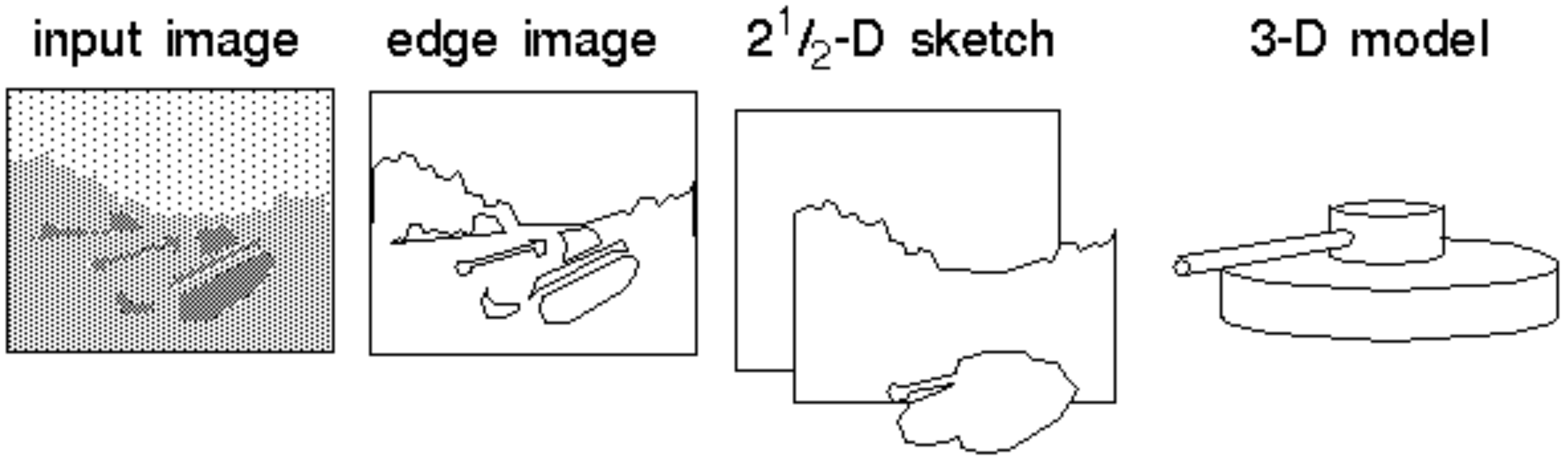
- **Synthetic** data
  - **Generative model** + photo-realistic **rendering**
  - Object geometry + physical **simulation**
  - Object geometry + synthetic materials + realistic simulations

- **Real** data
  - Collected from images, scans, mocap sessions
  - Collected using specialized equipments (e.g., light-field, pressure gloves)

End-to-end: Learned Features

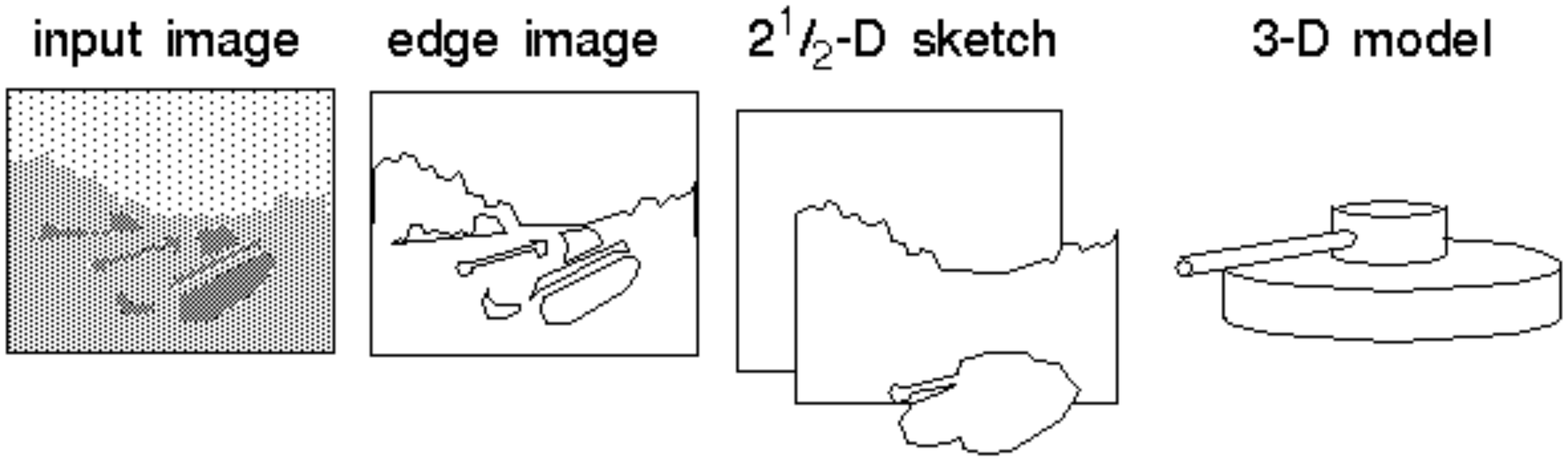
End-to-end: Learned Features

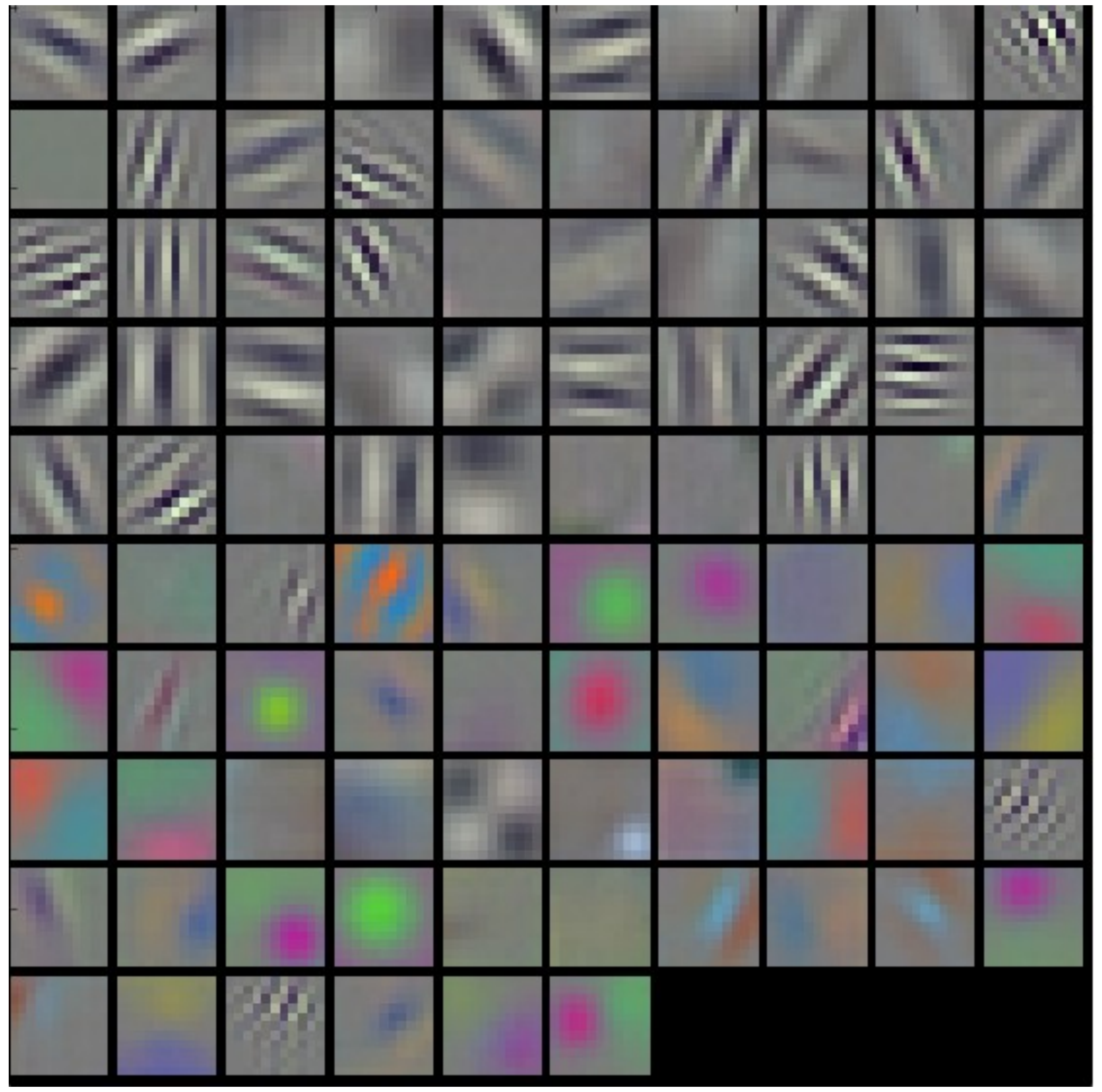
- *Before*
  - Handcrafted feature extraction, e.g., edges or corners (hand-crafted)
  - Mostly with linear models (PCA)
- *Now*



End-to-end: Learned Features

- *Before*
  - Handcrafted feature extraction, e.g., edges or corners (hand-crafted)
  - Mostly with linear models (PCA)
- *Now*
  - End-to-end
  - Move away from hand-crafted representations





End-to-end: Learned Loss

End-to-end: Learned Loss

- *Before*
  - Evaluation came after
  - It was a bit optional
    - You might still have a good algorithm without a good way of quantifying it
    - Evaluation helped publishing
- *Now*

End-to-end: Learned Loss

- *Before*
  - Evaluation came after
  - It was a bit optional
    - You might still have a good algorithm without a good way of quantifying it
    - Evaluation helped publishing
- *Now*
  - It is essential and build-in
  - If the loss is not good, the result is not good
  - (Extensive) Evaluation happens automatically

- While still much is left to do, this makes graphics much more reproducible

End-to-end Training: Real/Generated Data

End-to-end Training: Real/Generated **Data**

- *Before*
  - Test with some toy examples
  - Deploy on real stuff
  - Maybe collect some performance data later

- *Now*

End-to-end Training: Real/Generated Data

- *Before*
  - Test with some toy examples
  - Deploy on real stuff
  - Maybe collect some performance data later

- *Now*
  - Test and deploy need to be as identical **(in distribution)**
  - Need to collect data first
  - No two steps

Course Plan

- Understand **common ML methods** (supervised and unsupervised) used in CG
- Understand the **building blocks**
  - Commonly used architectures, loss function, training advice
- Opportunities to develop **new methods**
  - ML methods for CG-specific domains (e.g., points, meshes, graphs)
  - How to mix synthetic/real data (and distributions)

Code Examples

PCA/SVD basis
Linear Regression
Polynomial Regression
Stochastic Gradient Descent vs. Gradient Descent
Multi-layer Perceptron
Edge Filter 'Network'
Convolutional Network
Filter Visualization
Weight Initialization Strategies
Colorization Network
Autoencoder
Variational Autoencoder
Generative Adversarial Network

http://geometry.cs.ucl.ac.uk/creativeai/

Scan me

Other Courses at Siggraph 2019

- **Deep Learning: A Crash Course**
  Andrew Glassner
  Sunday 9:00-12:15

- **Geometric Computing with Python**
  Sebastian Koch, Teseo Schneider, Francis Williams, Daniele Panozzo
  Tuesday 2:00-3:30

- **Differential Graphics with Tensorflow**
  Sofien Bouaziz, Martin Wicke, Julien Valentin, Paige Bailey, Josh Gordon, Christian Haene, Alexander Mordvintsev, Shan Carter
  Thursday 9:00-12:15

Examples in Graphics

Geometry

Image manipulation

Animation

Rendering

Examples in Graphics

Geometry

Procedural modelling

Mesh segmentation

Learning deformations

Colorization

Sketch simplification

Image manipulation

Animation

BRDF estimation

Boxification

Fluid

Real-time rendering

Animation

Rendering

Denoising

Facial animation

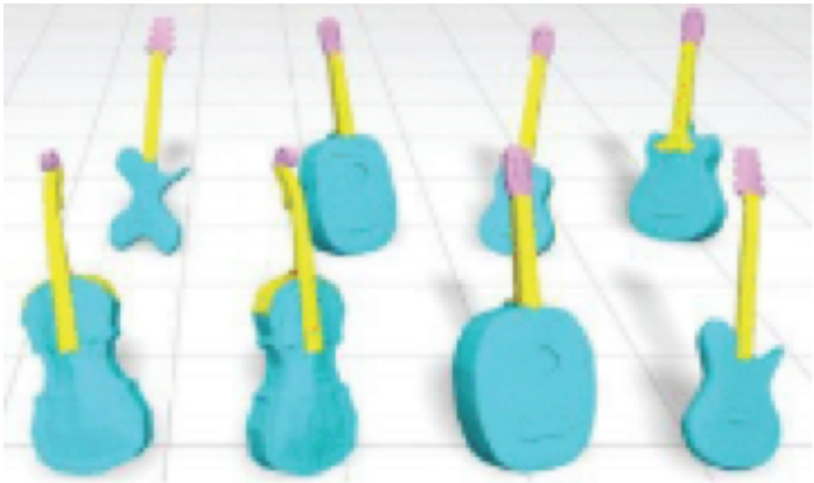
PCD processing

Examples in Graphics

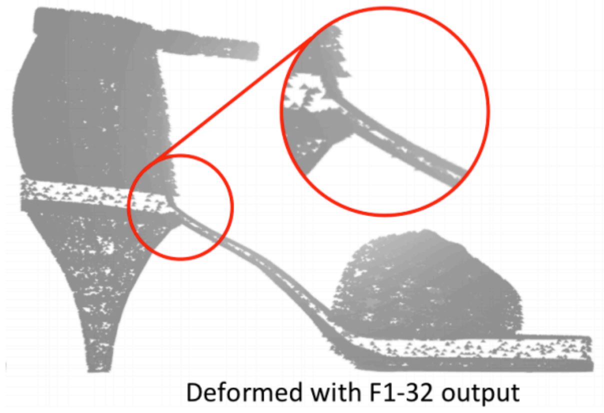
Sketch simplification

Colorization

Procedural modelling



Mesh segmentation

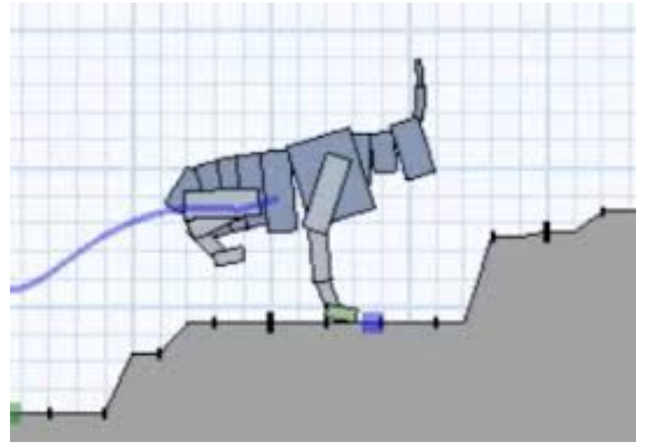


Learning deformations

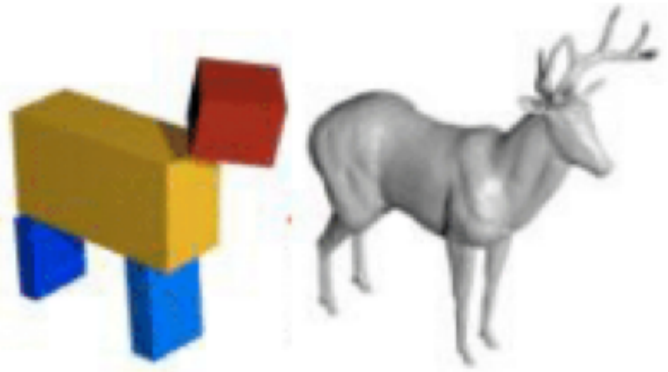
Real-time rendering

BRDF estimation

Fluid

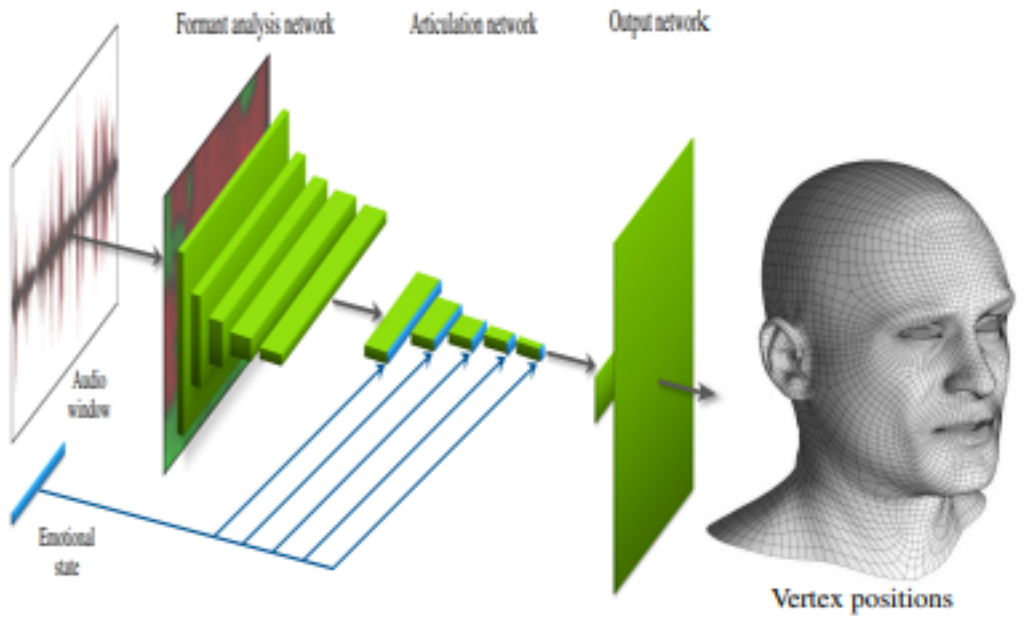


Animation

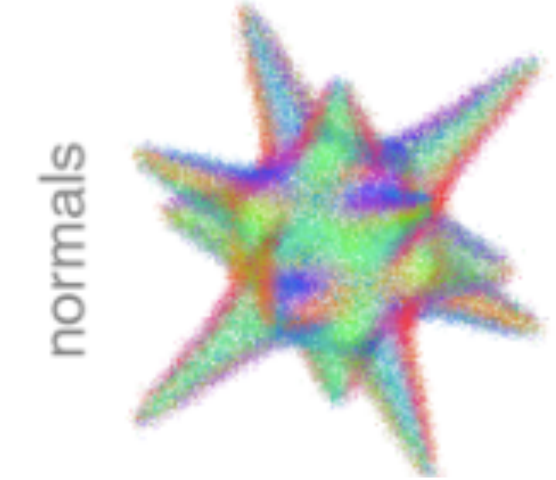


Boxification

Denoising

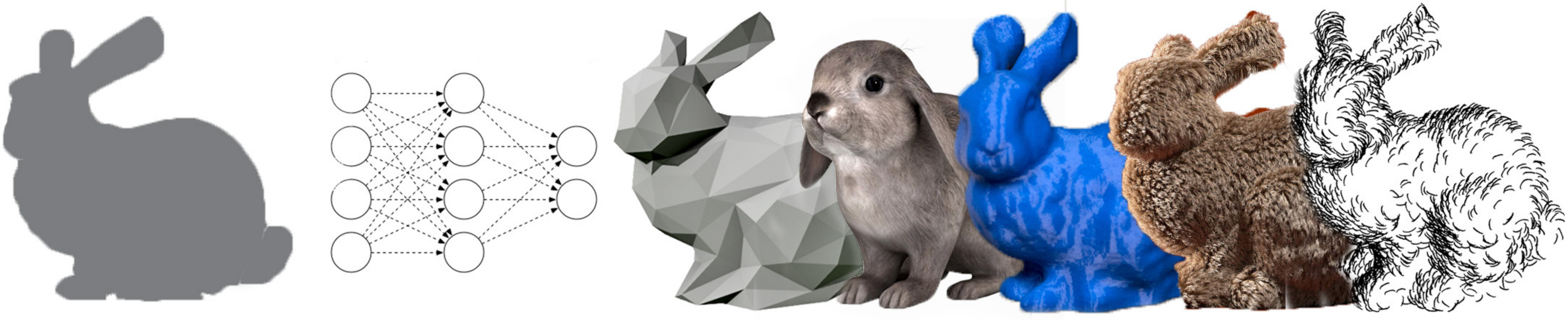


Facial animation



PCD processing

Course Information (slides/code/comments)



http://geometry.cs.ucl.ac.uk/creativeai/

Scan me