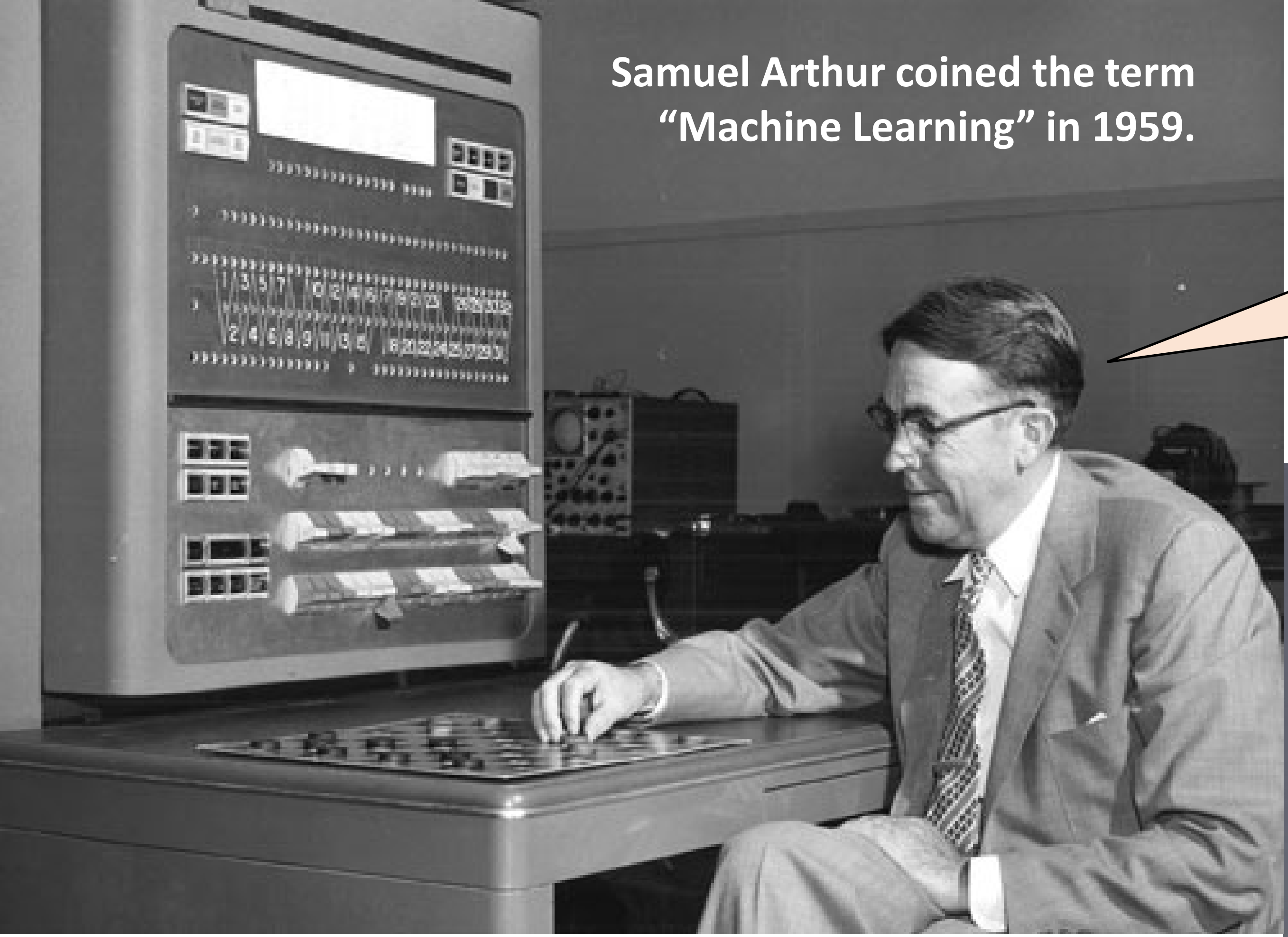
Introduction to Machine Learning

Hiranmayi Ranganathan, LLNL

June 19, 2020

Learning Objectives

- What is Machine Learning?
- Main branches of Machine Learning
- Machine Learning Workflow
  - Data Collection
  - Feature Engineering
  - Training
  - Validation
  - Testing
  - Evaluation, hyperparameter tuning
- Scikit-Learn Examples (independent learning)



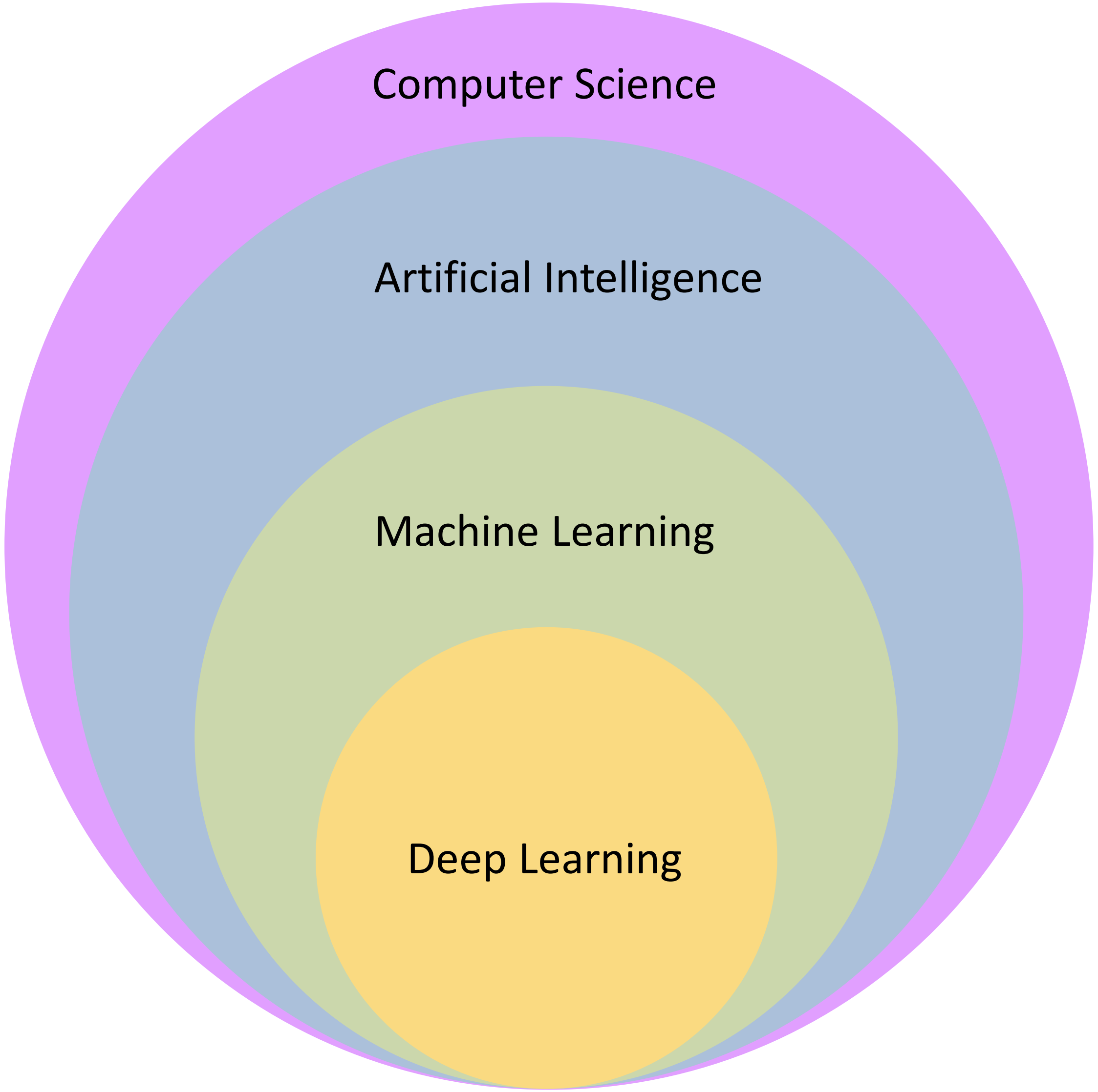
Samuel Arthur coined the term "Machine Learning" in 1959.

Field of study that gives computers the ability to learn without being explicitly programmed.

Tom Mitchell defined ML as a **well-posed learning problem** in 1998.

A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

AI ⊃ ML ⊃ DL



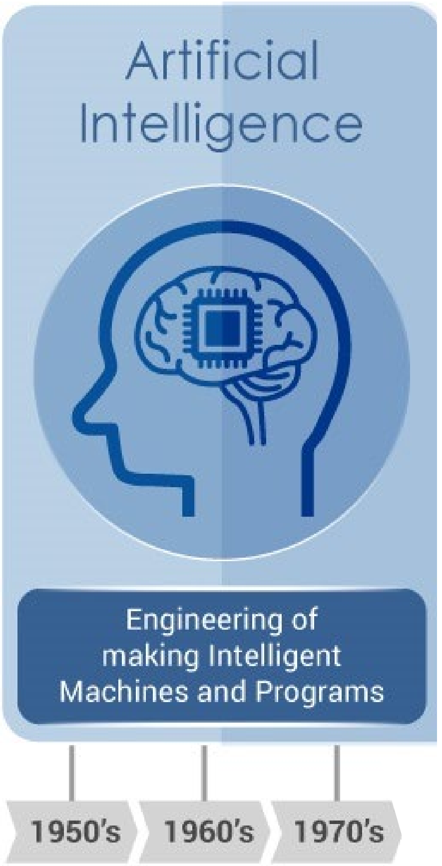


Image source: https://www.embedded-vision.com/industry-analysis/blog/artificial-intelligence-machine-learning-deep-learning-and-computer-visionwha

Artificial Intelligence

“This work is really tedious, yet requires a lot of troubleshooting and problem solving. Maybe I can get a machine to do it for me.”

Machine Learning

“It’s really difficult to program this computer to understand what I need it to do. Maybe it can teach itself how to do it, if I can help it along by distilling the data into the meaningful examples and exposing enough of these examples to the computer.

Deep Learning

“I don’t know the best way to distill the data into meaningful examples. Maybe if I can give it TONS of data, it can figure out what’s important from the data without my help.

Adopted from: https://medium.com/eliza-effect/ai-ml-and-deep-learning-whats-the-difference-ba6bb2223589

Artificial Intelligence

1950 – Alan Turing proposes his Turing Test for intelligence.

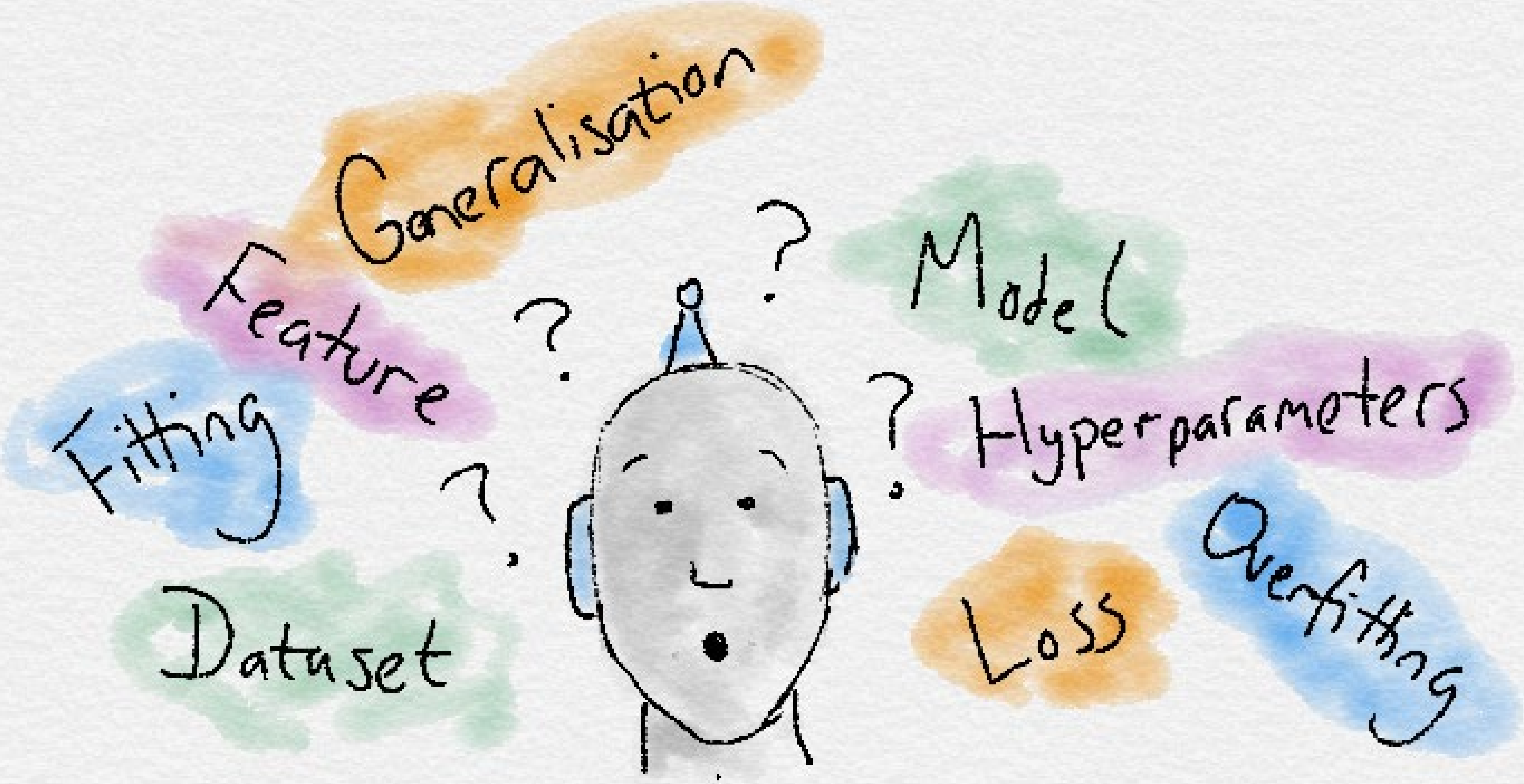
Machine Learning

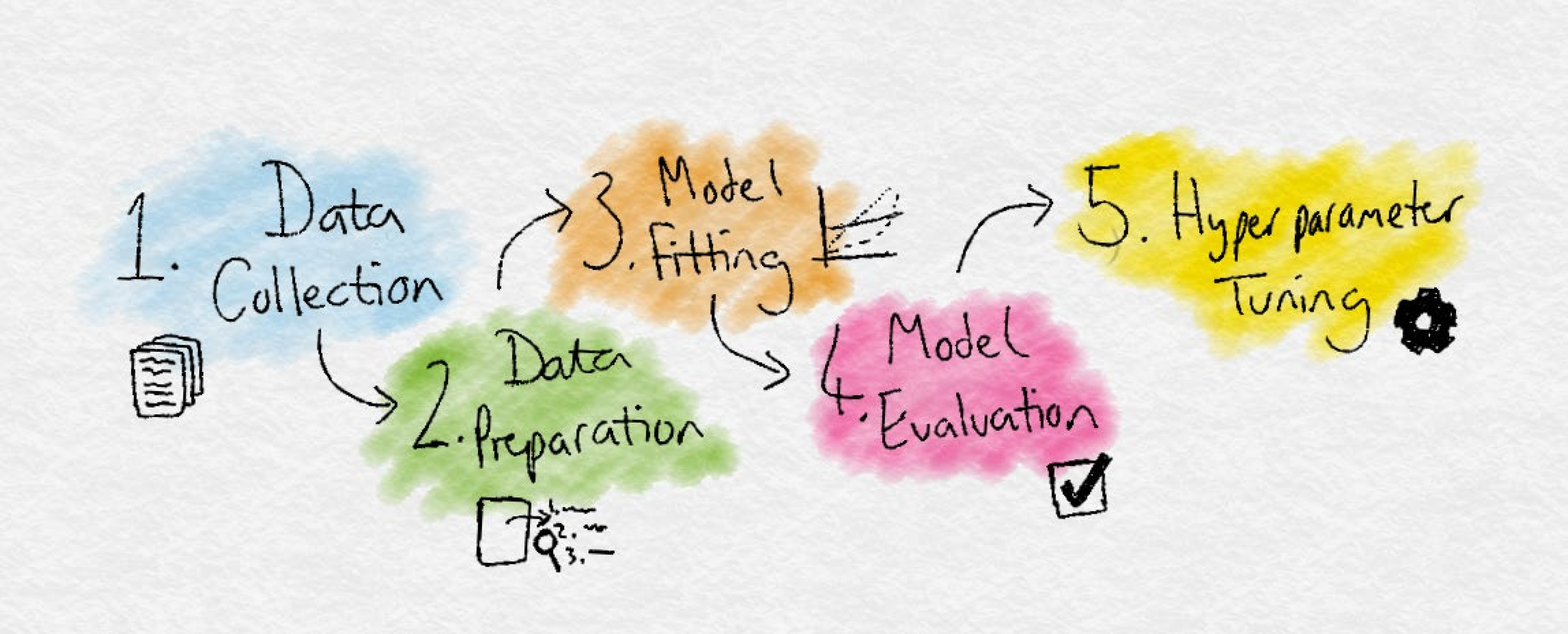
1952 – Arthur Samuel wrote the first computer learning program.
1957 – Frank Rosenblatt designed the first neural network.

Deep Learning

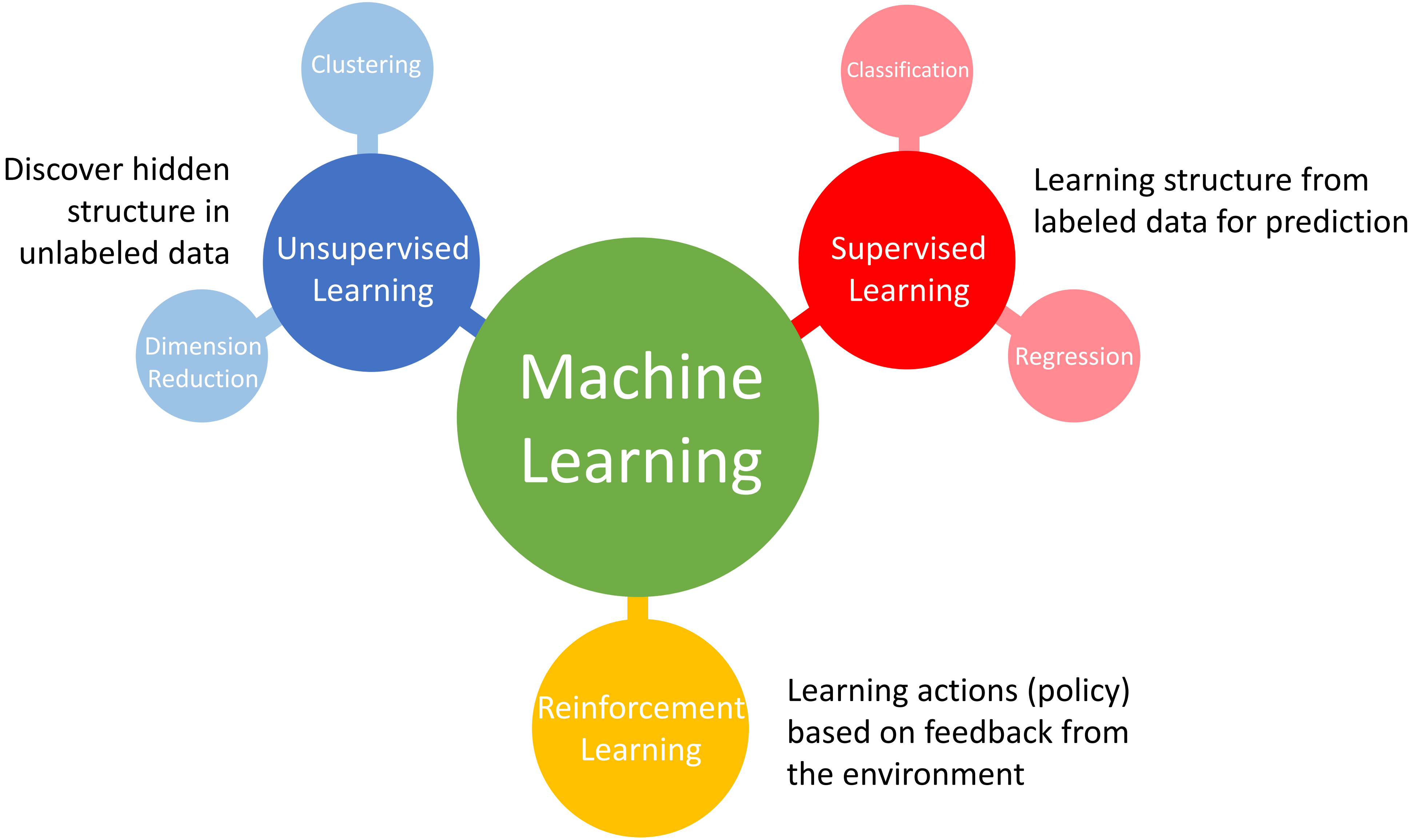
2006 – Geoffrey Hinton coins the term “Deep Learning”.
2012 – Google is able to identify videos that contain cats.
2014 – Facebook can verify people in photos.
2016 – Google’s AlphaGo beats world champion at Go.

Image source: https://medium.com/eliza-effect/ai-ml-and-deep-learning-whats-the-difference-ba6bb2223589


Generalisation
Feature
Fitting
Dataset
Model
Hyperparameters
Loss
Overfitting


1. Data Collection
2. Data Preparation
3. Model Fitting
4. Model Evaluation
5. Hyperparameter Tuning

Machine Learning Approaches



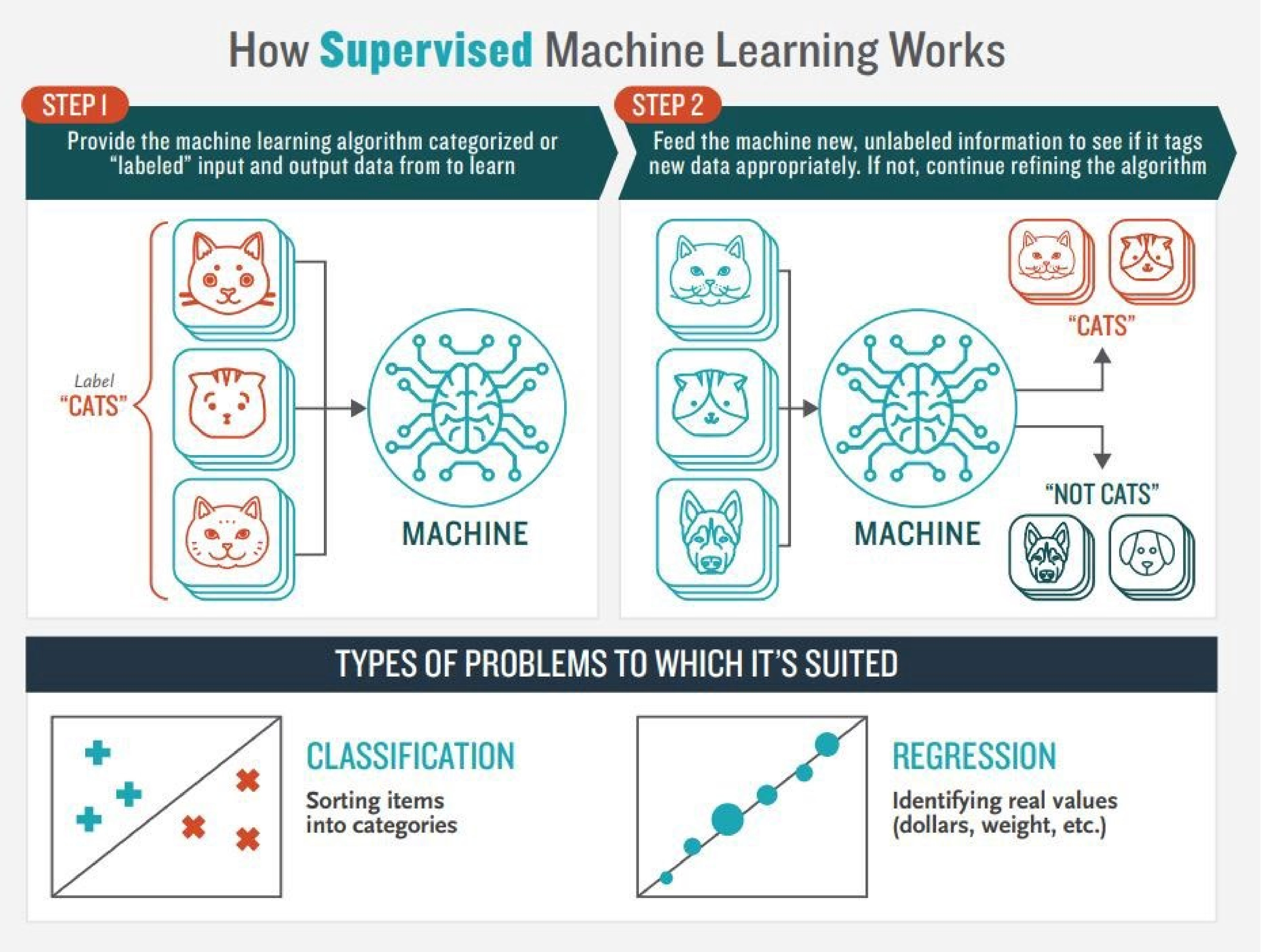


Image source: boozallen.com

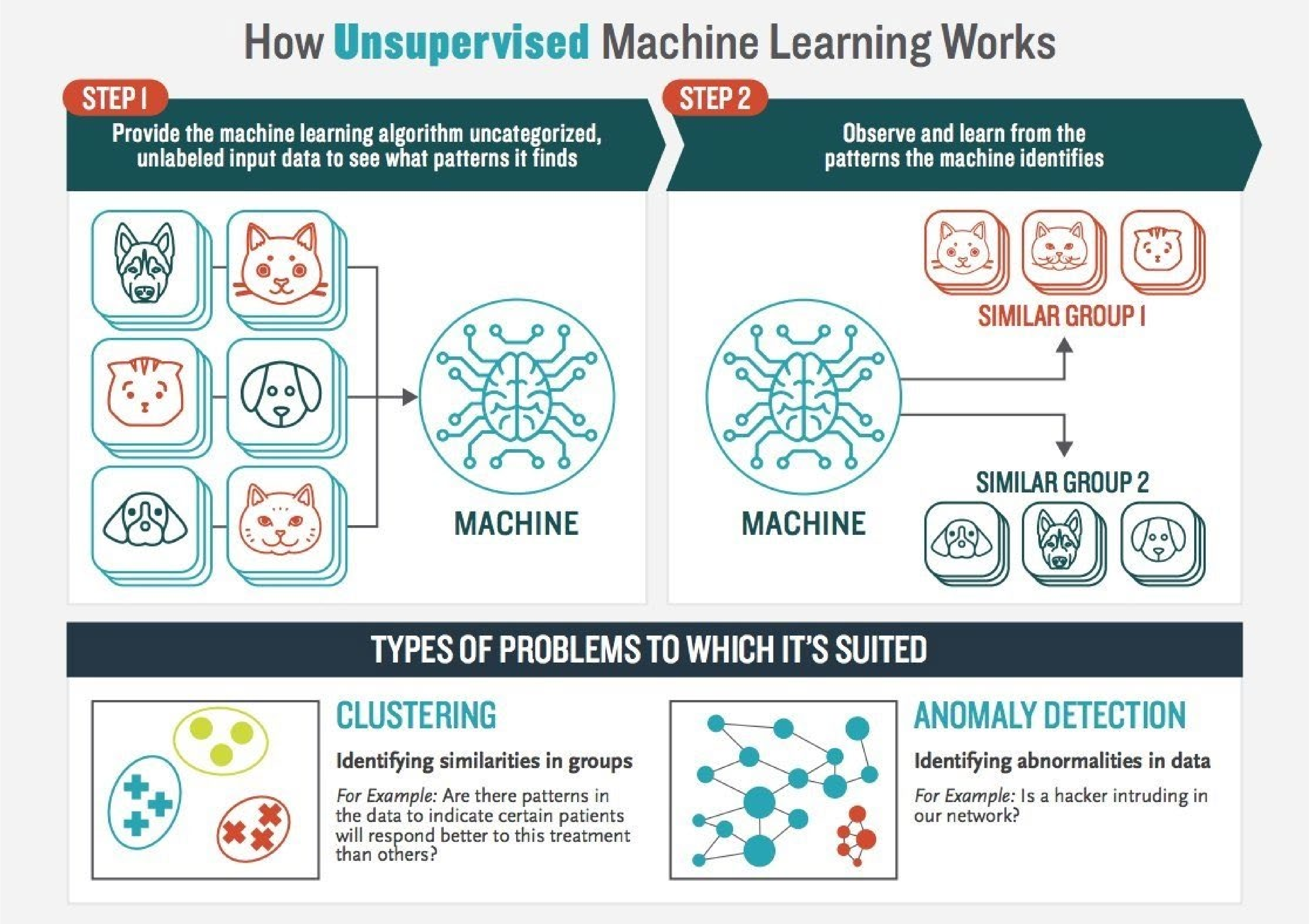


Image source: boozallen.com

Overview of models under categories

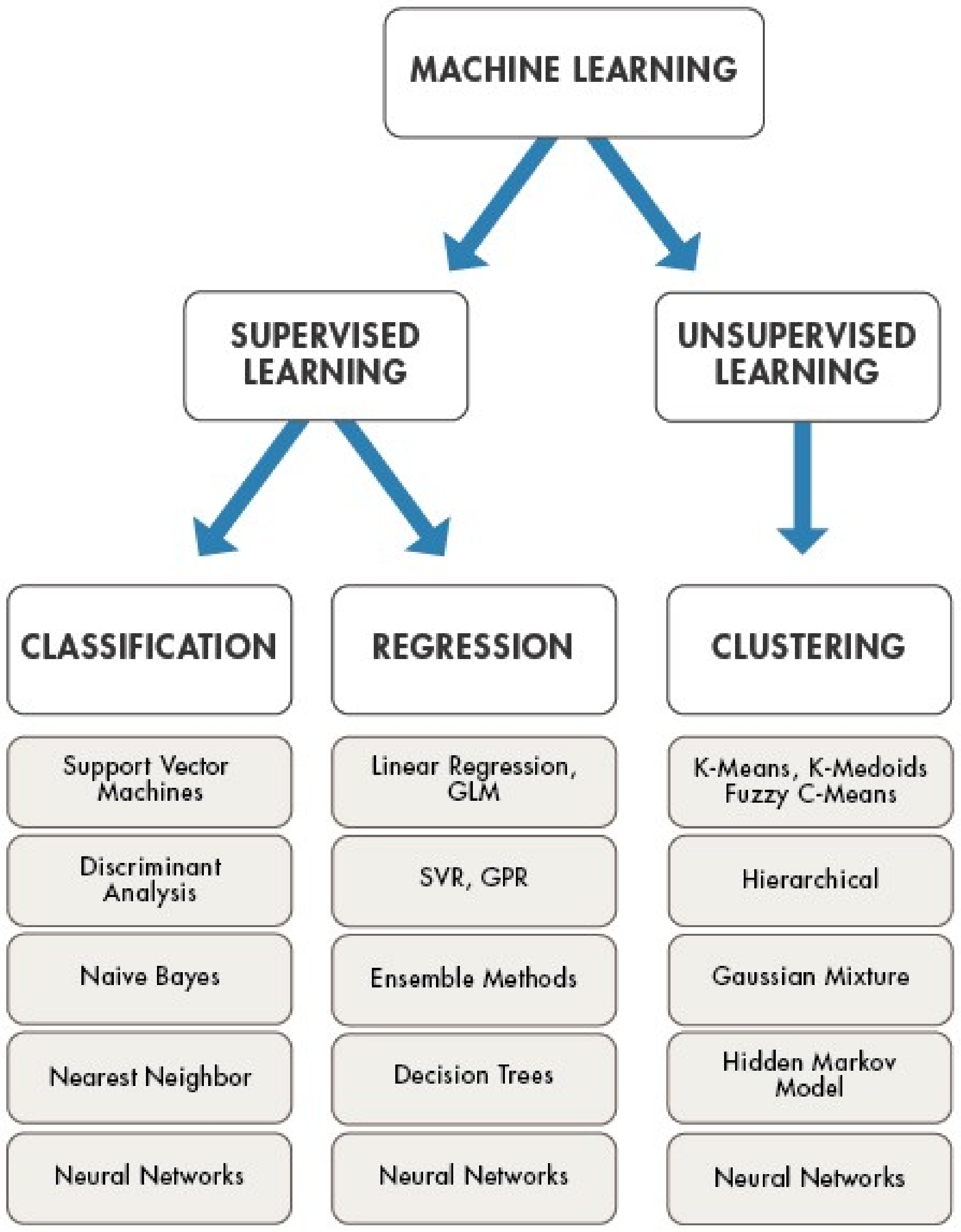
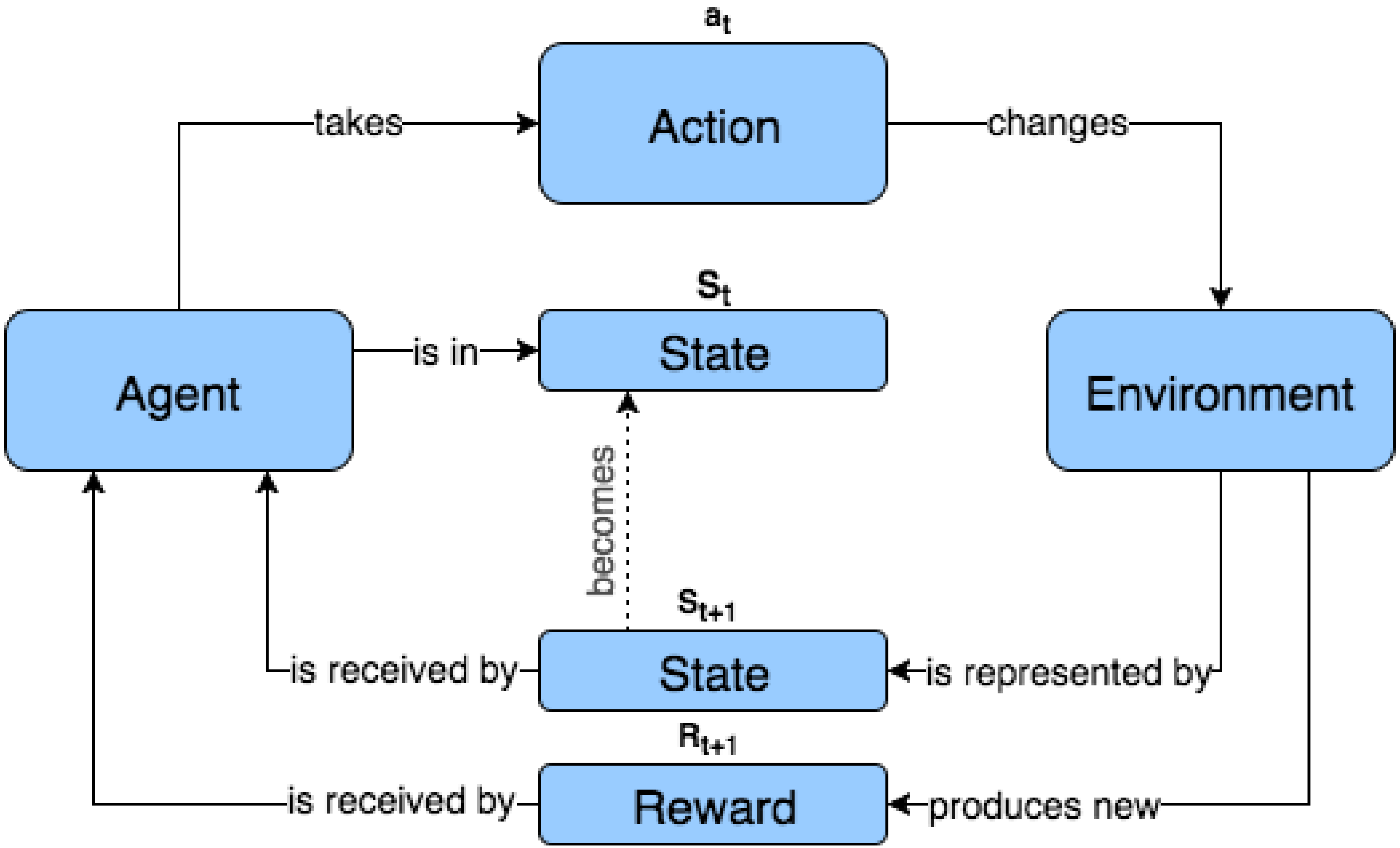


Image source: https://livebook.manning.com/#!/book/deep-reinforcement-learning-in-action/chapter-1/v-4/47

How reinforcement learning works

The agent's job is to take actions to maximize its cumulative rewards.



The reward is a time-delayed feedback on the optimality of the agent's action.

The ML workflow

1. **Define the Problem appropriately**
2. **Collect Data**
3. **Choose a Measure of Success**

The ML workflow

1. **Define Appropriately the Problem**
2. **Collect Data**
3. **Choose a Measure of Success:**
4. **Preparing The Data**

Methods of Preparing The Data

- **Dealing with missing data**

- **Handling Caterogical Data**

Color
Red
Red
Yellow
Green
Yellow

→

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

- **Feature Scaling**

$$X_{changed} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

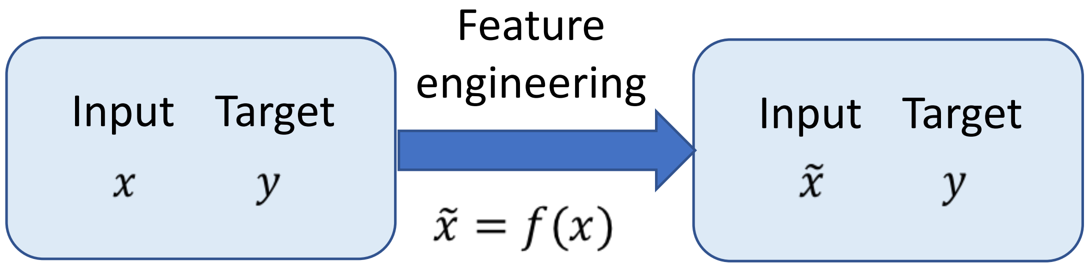
$$z = \frac{x - \mu}{\sigma}$$

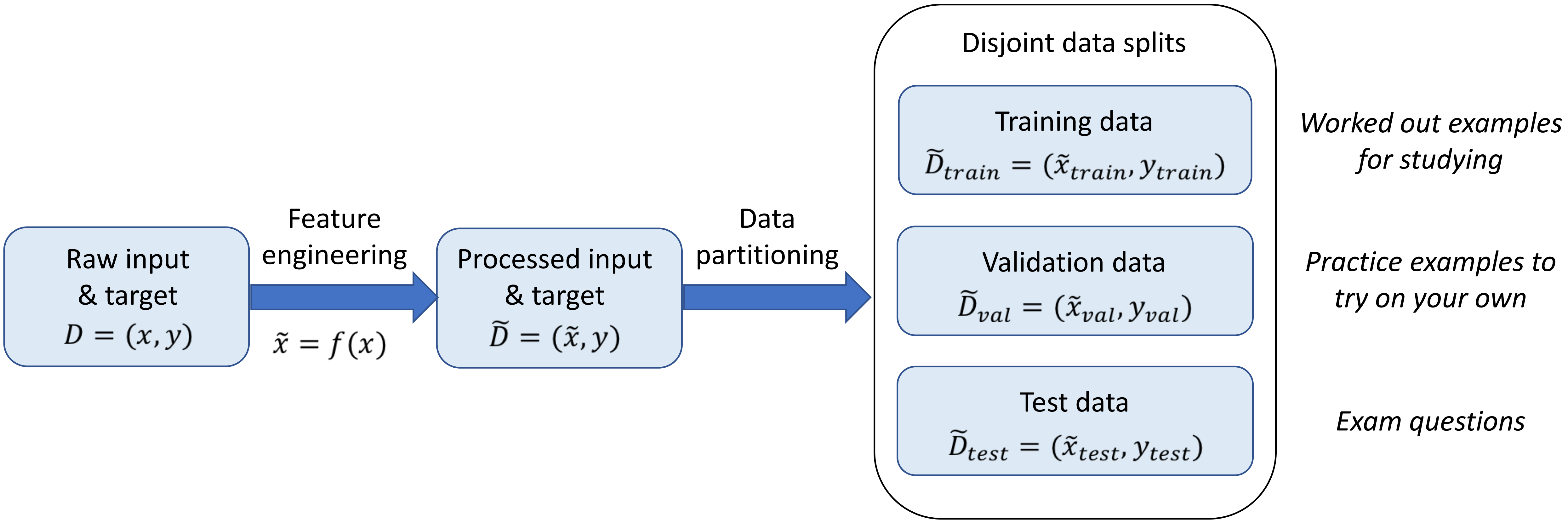
$\mu =$ Mean

$\sigma =$ Standard Deviation

The ML workflow

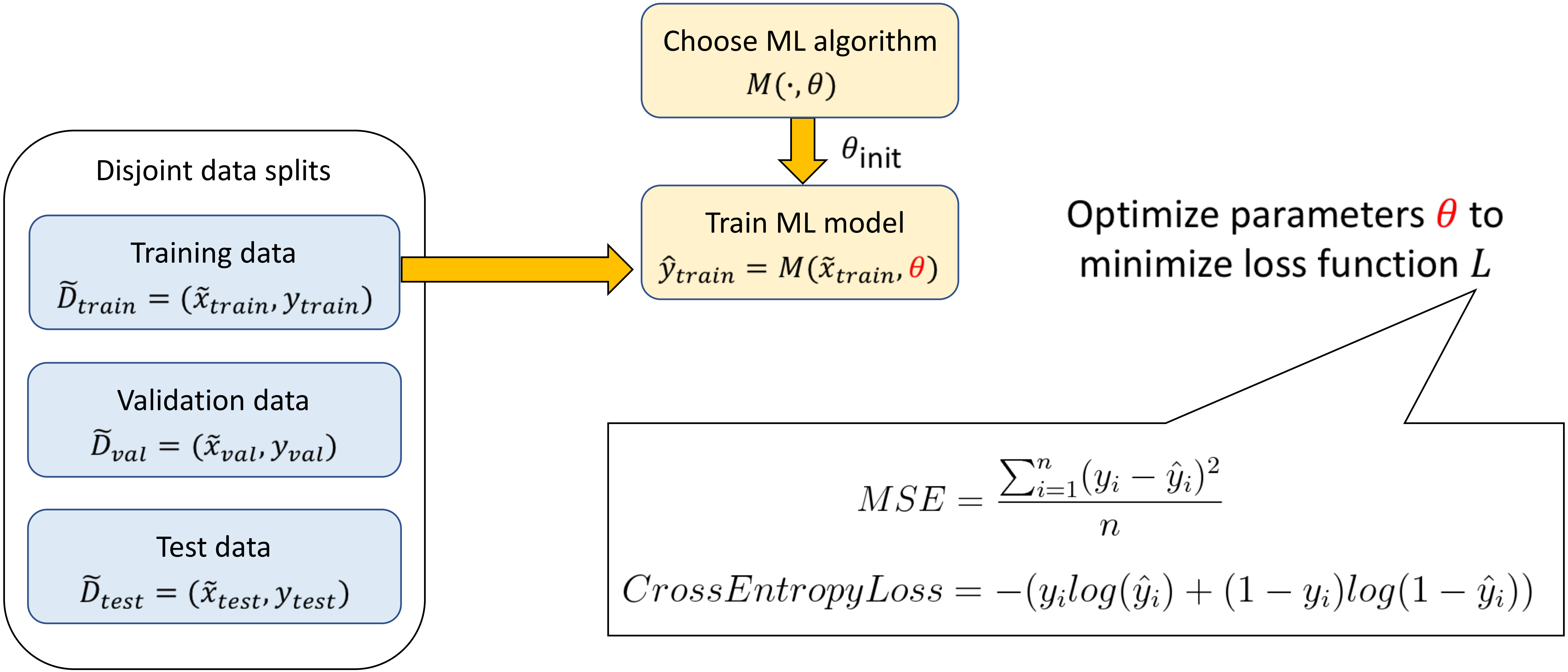
1. **Define Appropriately the Problem**
2. **Collect Data**
3. **Choose a Measure of Success:**
4. **Preparing The Data**
5. **Splitting Data Into Subsets**


Input
Target
x
y
Feature engineering
$\tilde{x} = f(x)$
Input
Target
$\tilde{x}$
y


Raw input
& target
$D = (x, y)$
Feature
engineering
$\tilde{x} = f(x)$
Processed input
& target
$\tilde{D} = (\tilde{x}, y)$
Data
partitioning
Disjoint data splits
Training data
$\tilde{D}_{train} = (\tilde{x}_{train}, y_{train})$
Validation data
$\tilde{D}_{val} = (\tilde{x}_{val}, y_{val})$
Test data
$\tilde{D}_{test} = (\tilde{x}_{test}, y_{test})$
Worked out examples for studying
Practice examples to try on your own
Exam questions

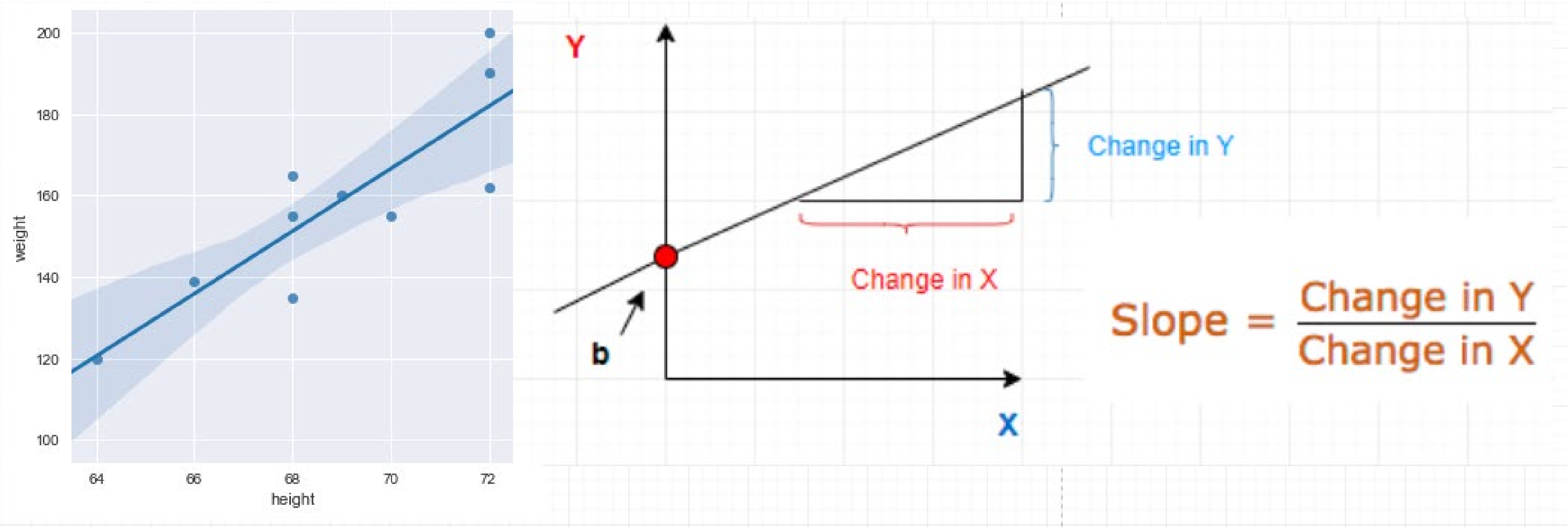
The ML workflow

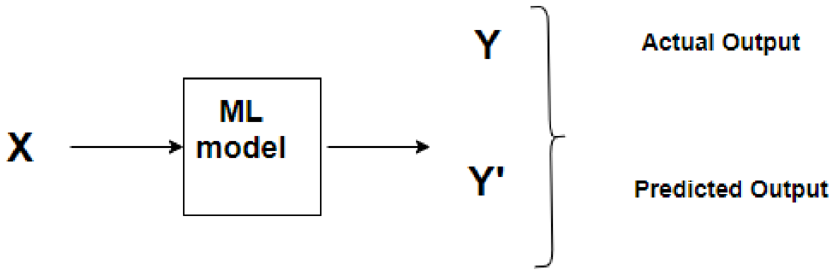
1. **Define Appropriately the Problem**
2. **Collect Data**
3. **Choose a Measure of Success:**
4. **Preparing The Data**
5. **Splitting Data Into Subsets**
6. **Training the Model**



Disjoint data splits
Training data
$\widetilde{D}_{train} = (\tilde{x}_{train}, y_{train})$
Validation data
$\widetilde{D}_{val} = (\tilde{x}_{val}, y_{val})$
Test data
$\widetilde{D}_{test} = (\tilde{x}_{test}, y_{test})$
Choose ML algorithm
$M(\cdot, \theta)$
θ_{init}
Train ML model
$\hat{y}_{train} = M(\tilde{x}_{train}, \theta)$
Optimize parameters θ to minimize loss function L
$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$
$CrossEntropyLoss = -(y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i))$

A Machine Learning Model





Error = Y'(Predicted) - Y(Actual)

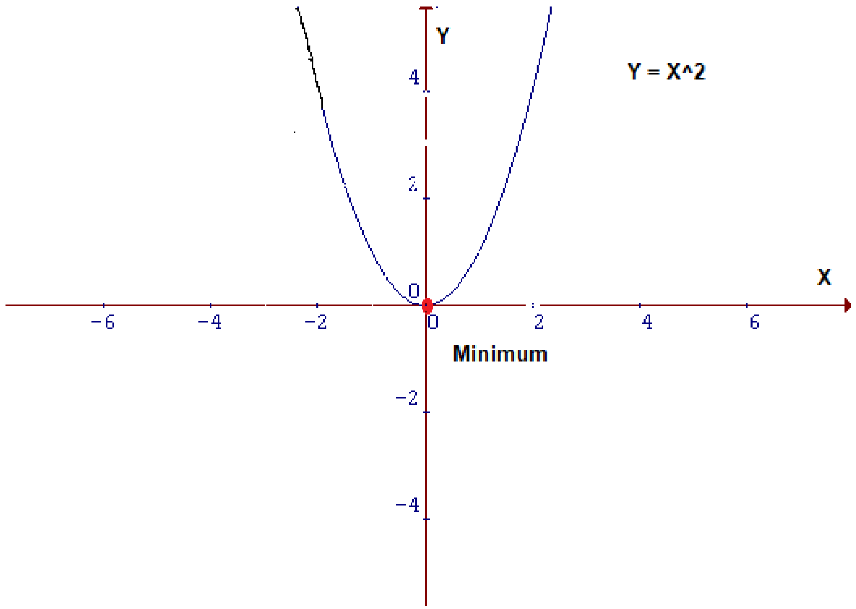
Cost function

Let's say, there are a total of 'N' points in the dataset and for all those 'N' data points we want to minimize the error. So the Cost function would be the total squared error i.e

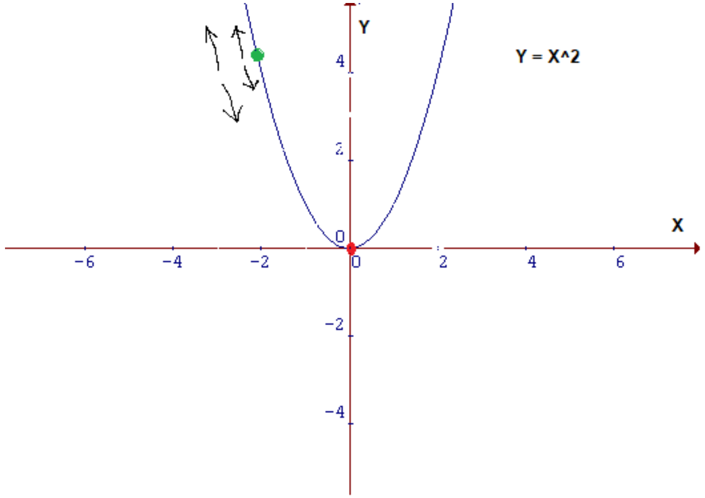
$$Cost = \frac{1}{N}\sum_{i=1}^{N}(Y' - Y)^2$$

How do we actually minimize any function?

Cost function is of the form **Y = X²**



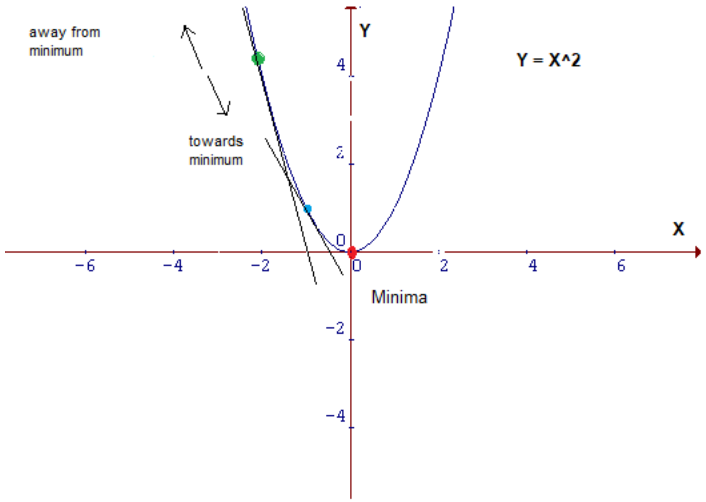
Gradient Descent



Possible actions would be:
•You might go upward or downward
•If you decide on which way to go, you might take a bigger step or a little step to reach your destination.

Essentially, there are two things that you should know to reach the minima, i.e. which way to go and how big a step to take.

The Minimum Value



The slope at the blue point is less steep than that at the green point which means it will take much smaller **steps** to reach the minimum from the blue point than from the green point.

Mathematical Interpretation of Cost Function

Parameters with small changes:

$$m = m - \delta m$$
$$b = b - \delta b$$

Given Cost Function for 'N' no of samples

$$Cost = \frac{1}{N} \sum_{i=1}^{N} (Y_i' - Y_i)^2$$

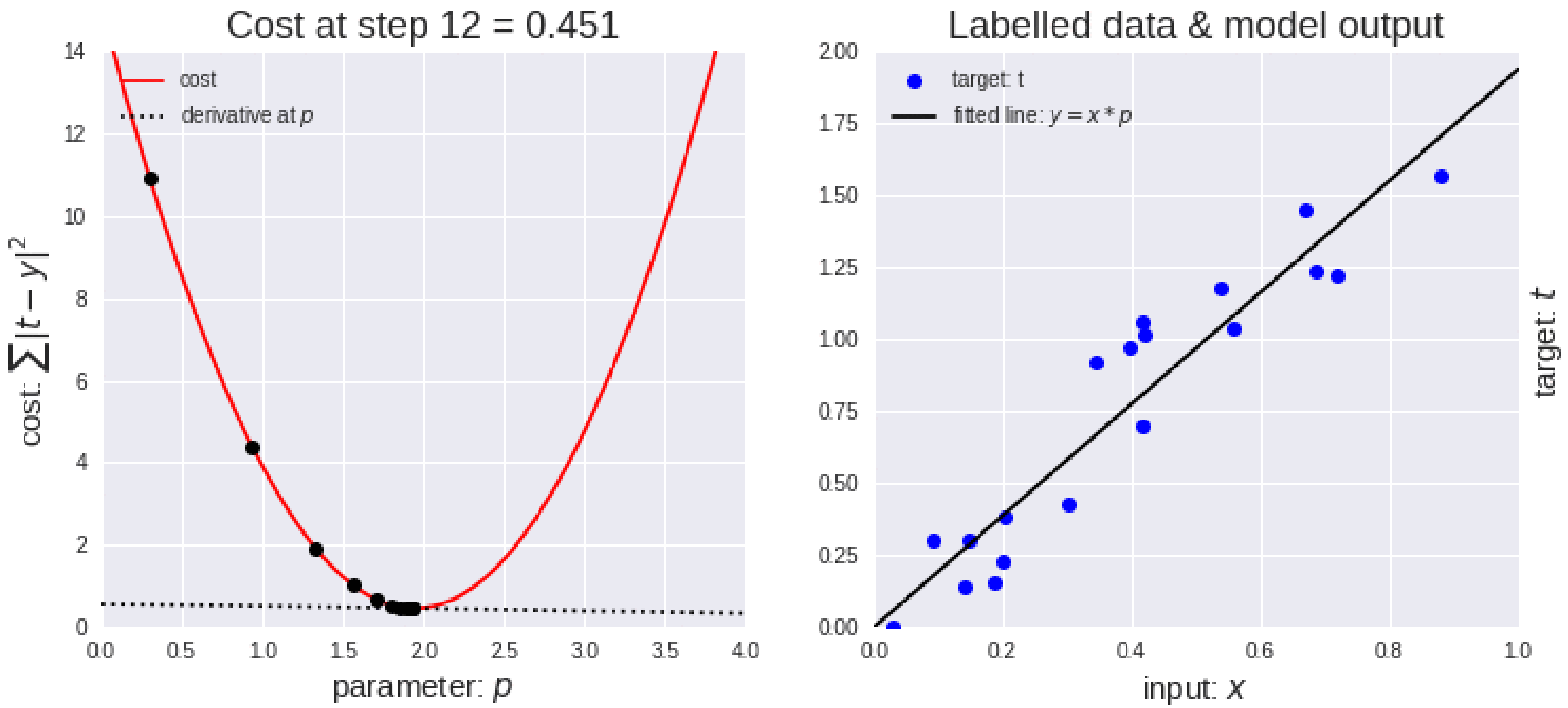
Cost function is denoted by J where J is a function of m and b

$$J_{m,b} = \frac{1}{N} \sum_{i=1}^{N} (Y_i' - Y_i)^2$$

Substituting the term Y'-Y with error for simplicity

$$J_{m,b} = \frac{1}{N} \sum_{i=1}^{N} (Error_i)^2$$

Gradient Descent



Disjoint data splits

Training data
$\widetilde{D}_{train} = (\tilde{x}_{train}, y_{train})$

Validation data
$\widetilde{D}_{val} = (\tilde{x}_{val}, y_{val})$

Test data
$\widetilde{D}_{test} = (\tilde{x}_{test}, y_{test})$

Choose ML algorithm
$M(\cdot, \theta)$

θ_{init}

Train ML model
$\hat{y}_{train} = M(\tilde{x}_{train}, \theta)$

Optimize parameters θ to minimize loss function L

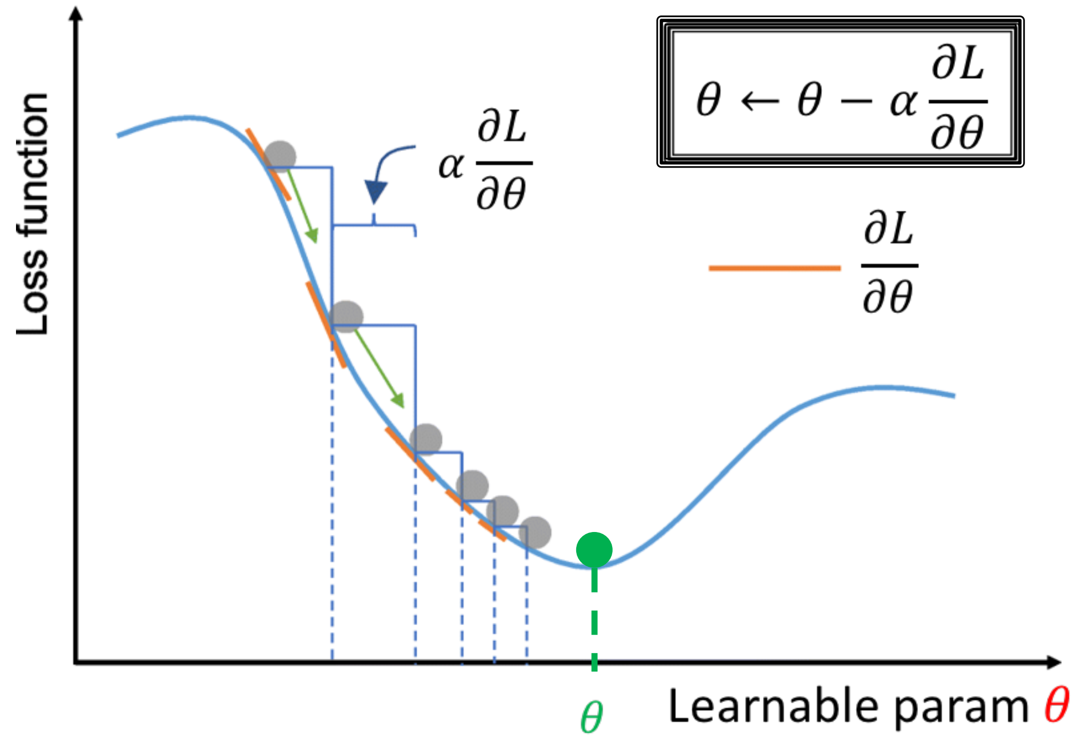


Image adopted from Yamashita et al. (2018)

Disjoint data splits

Training data
$\widetilde{D}_{train} = (\tilde{x}_{train}, y_{train})$

Validation data
$\widetilde{D}_{val} = (\tilde{x}_{val}, y_{val})$

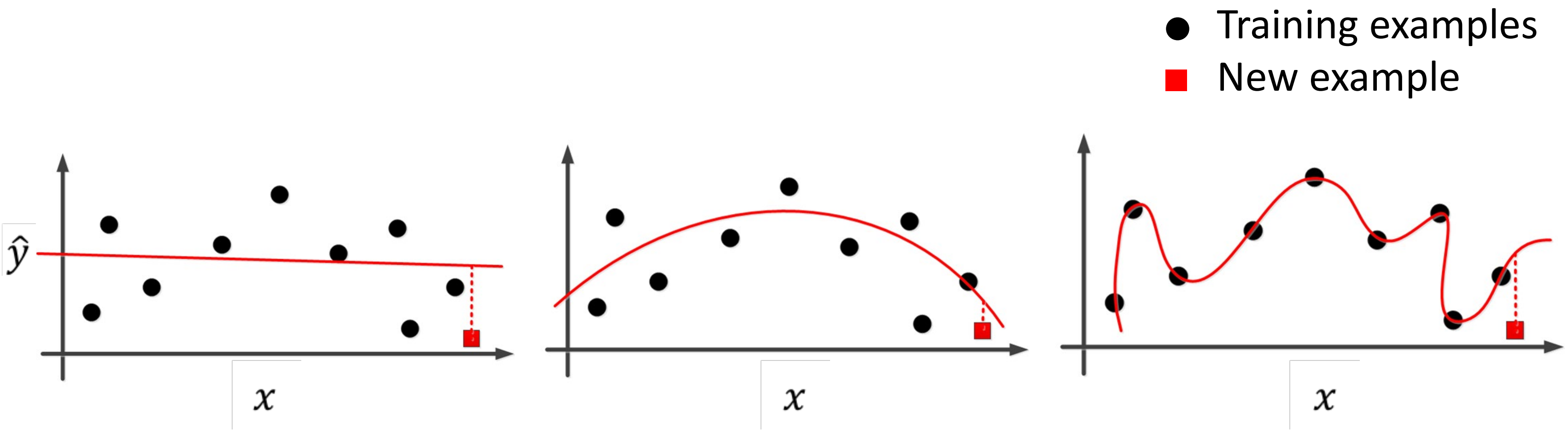
Test data
$\widetilde{D}_{test} = (\tilde{x}_{test}, y_{test})$

Choose ML algorithm
$M(\cdot, \theta)$

θ_{init}

Train ML model
$\hat{y}_{train} = M(\tilde{x}_{train}, \theta)$

After training, if we freeze the parameters, how good is the model?

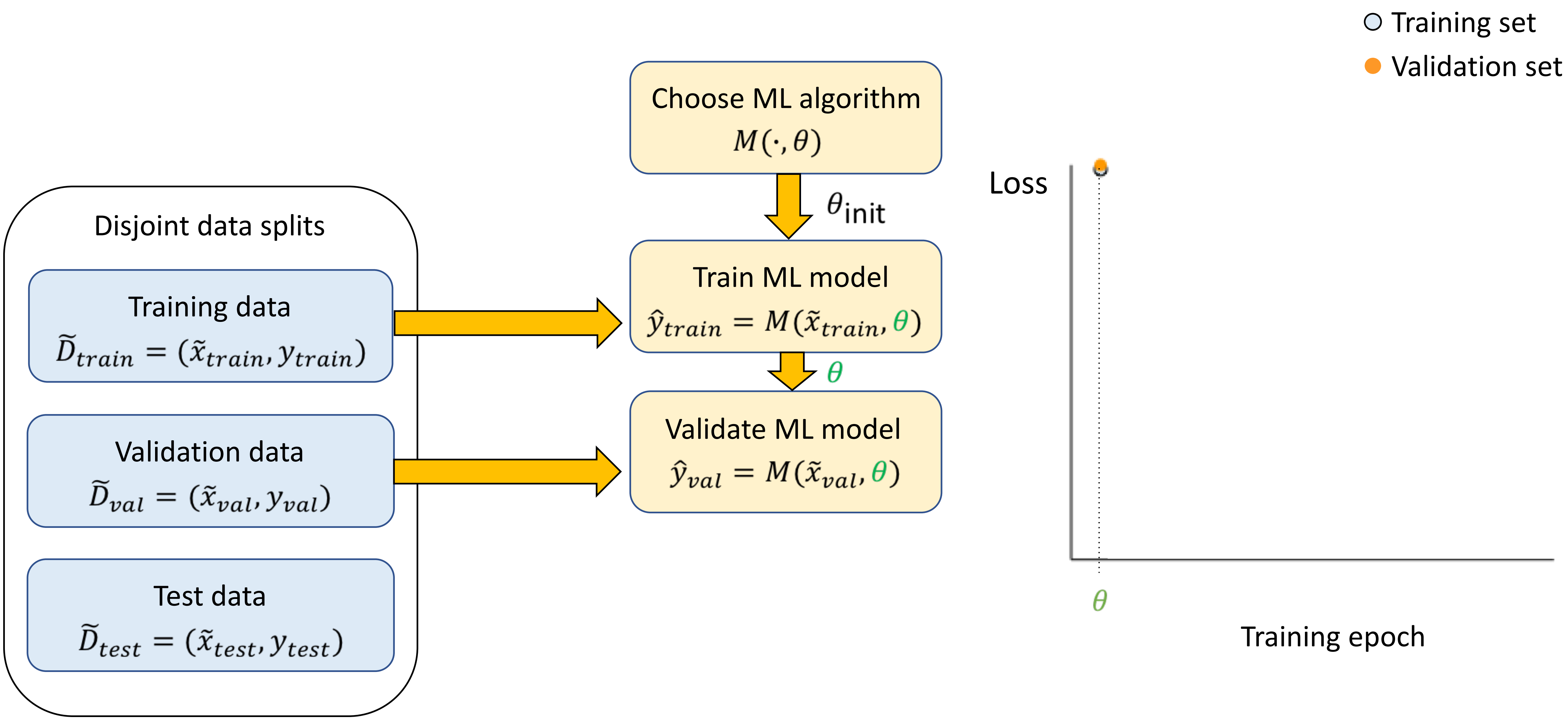


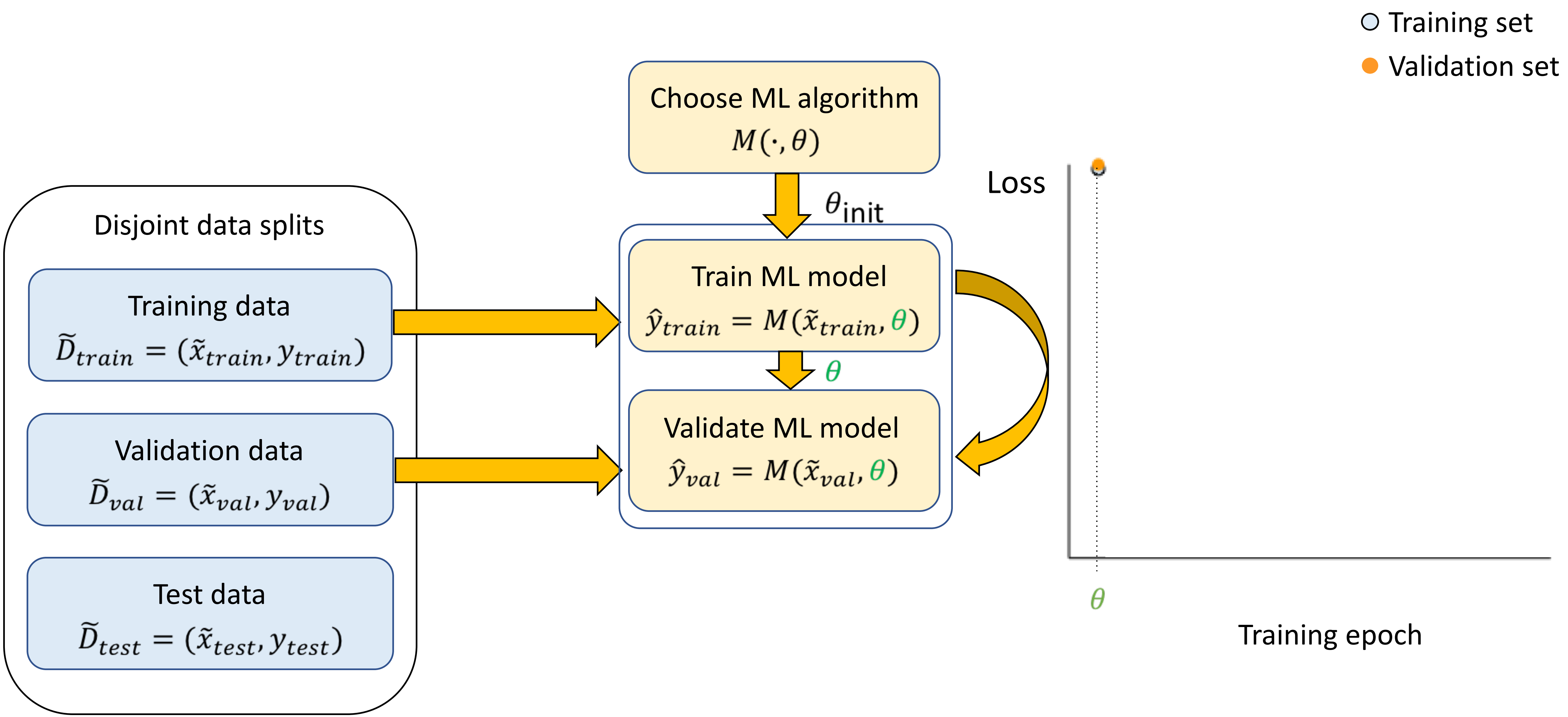
Underfitting

Good fit!

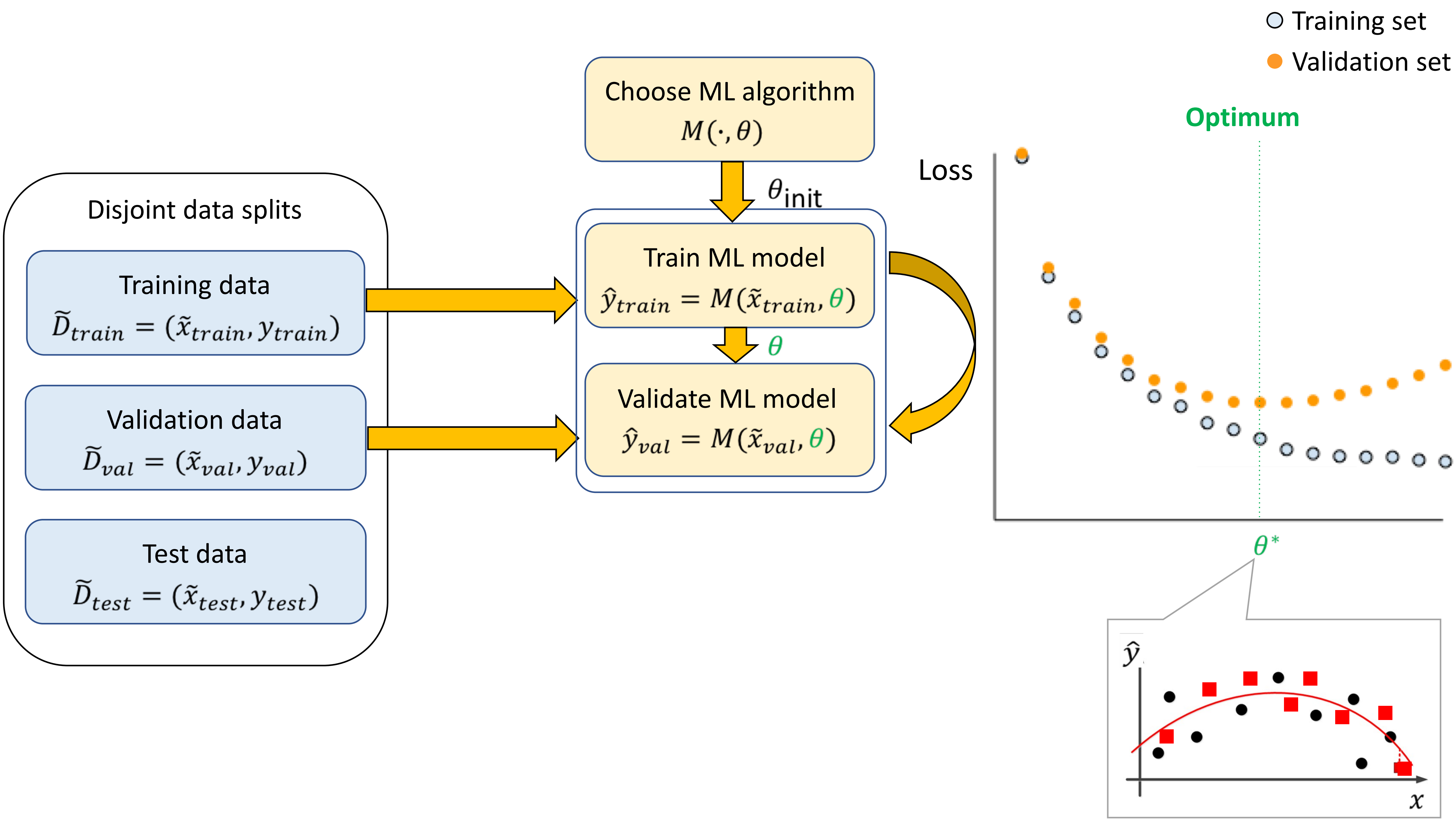
Overfitting

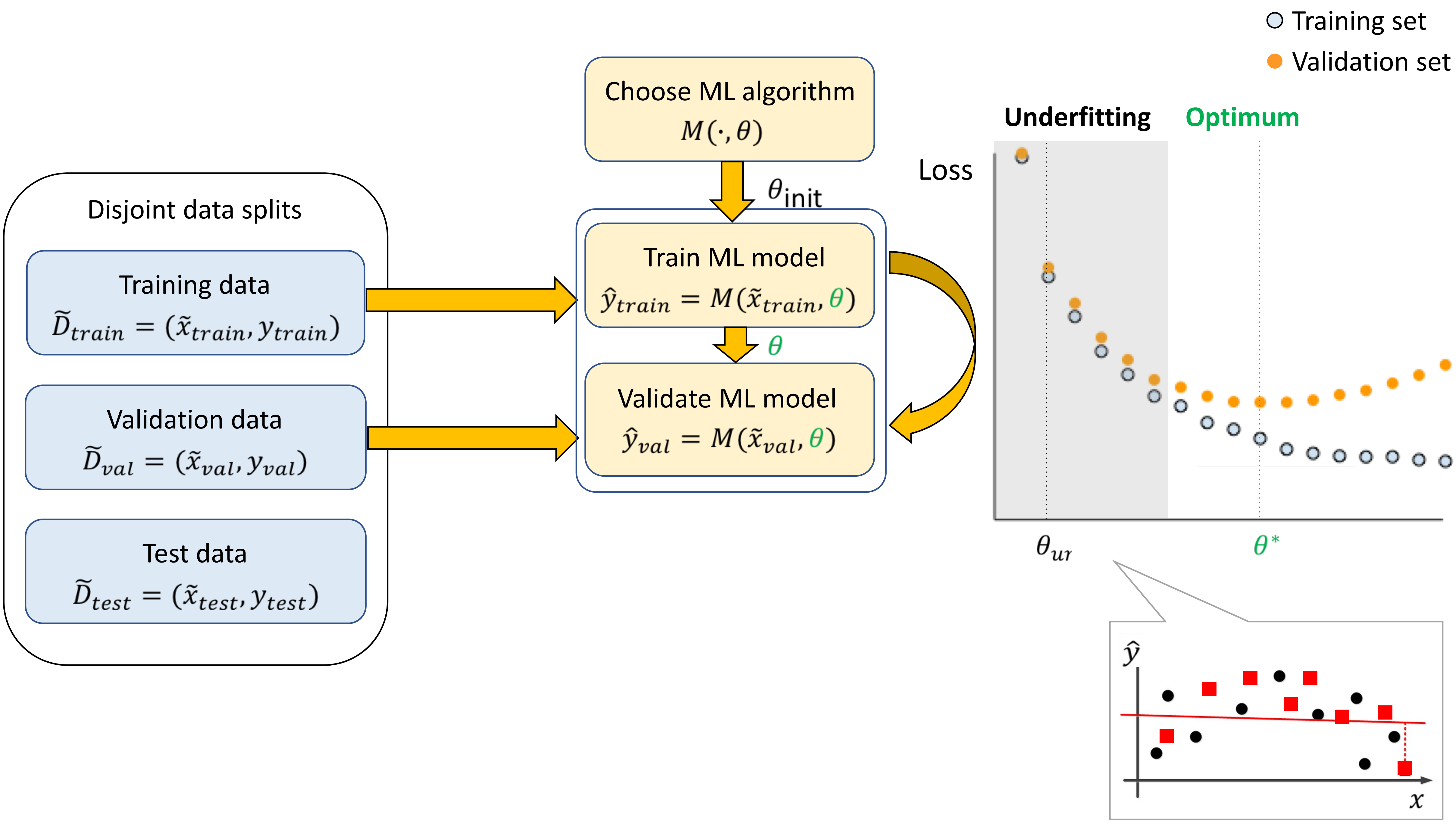
Image source: https://deeplearn.org/arxiv/77795/the-theory-behind-overfitting,-cross-validation,-regularization,-bagging,-and-boosting:-tutorial


Disjoint data splits
Training data
$\widetilde{D}_{train} = (\tilde{x}_{train}, y_{train})$
Validation data
$\widetilde{D}_{val} = (\tilde{x}_{val}, y_{val})$
Test data
$\widetilde{D}_{test} = (\tilde{x}_{test}, y_{test})$
Choose ML algorithm
$M(\cdot, \theta)$
θ_{init}
Train ML model
$\hat{y}_{train} = M(\tilde{x}_{train}, \theta)$
θ
Validate ML model
$\hat{y}_{val} = M(\tilde{x}_{val}, \theta)$
Training set
Validation set
Loss
θ
Training epoch

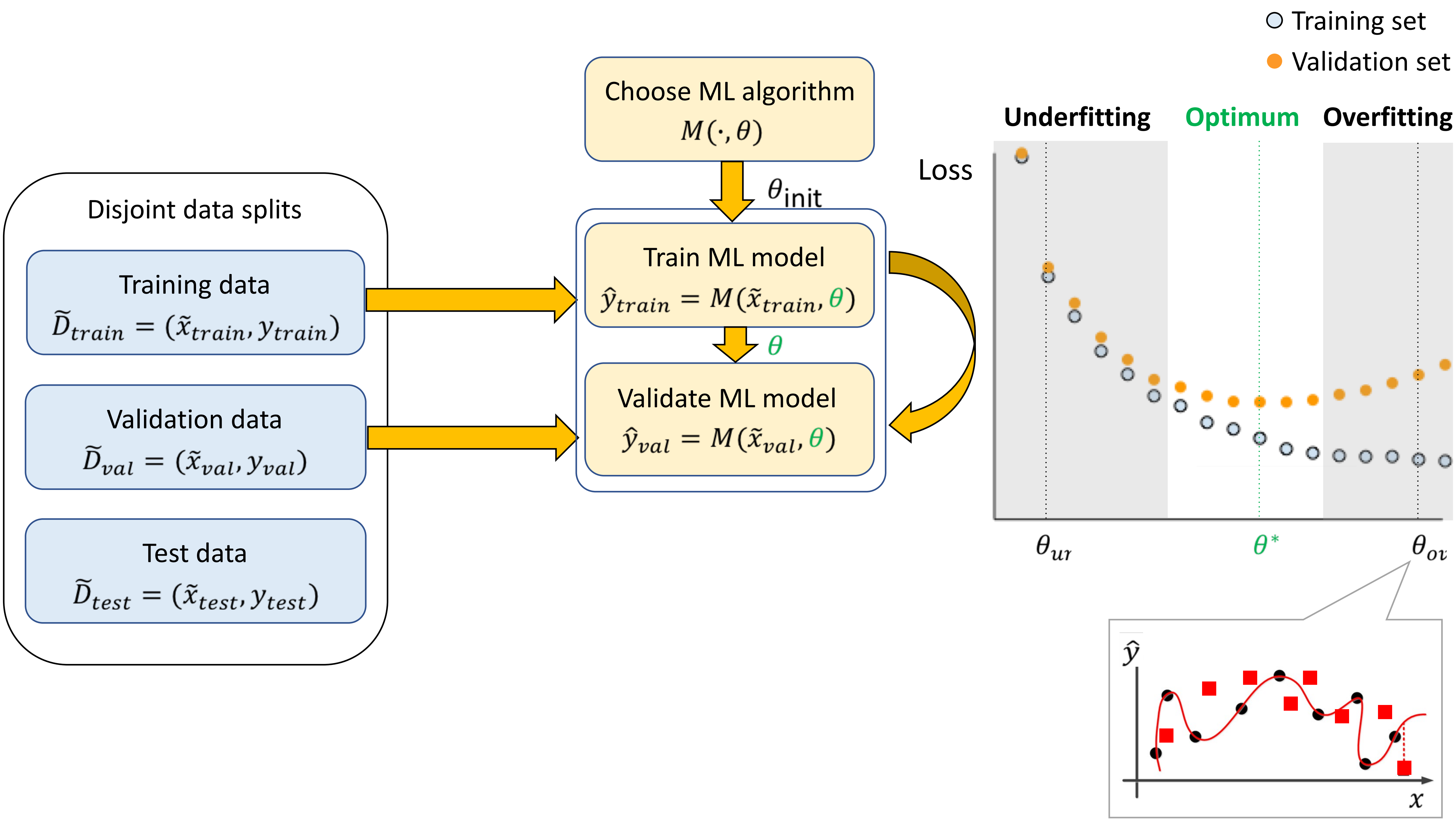


Training set
Validation set
Disjoint data splits
Training data
$\widetilde{D}_{train} = (\tilde{x}_{train}, y_{train})$
Validation data
$\widetilde{D}_{val} = (\tilde{x}_{val}, y_{val})$
Test data
$\widetilde{D}_{test} = (\tilde{x}_{test}, y_{test})$
Choose ML algorithm
$M(\cdot, \theta)$
θ_{init}
Train ML model
$\hat{y}_{train} = M(\tilde{x}_{train}, \theta)$
θ
Validate ML model
$\hat{y}_{val} = M(\tilde{x}_{val}, \theta)$
Loss
θ
Training epoch


Disjoint data splits
Training data
$\widetilde{D}_{train} = (\tilde{x}_{train}, y_{train})$
Validation data
$\widetilde{D}_{val} = (\tilde{x}_{val}, y_{val})$
Test data
$\widetilde{D}_{test} = (\tilde{x}_{test}, y_{test})$
Choose ML algorithm
$M(\cdot, \theta)$
θ_{init}
Train ML model
$\hat{y}_{train} = M(\tilde{x}_{train}, \theta)$
θ
Validate ML model
$\hat{y}_{val} = M(\tilde{x}_{val}, \theta)$
Training set
Validation set
Optimum
Loss
θ^*
$\hat{y}$
x



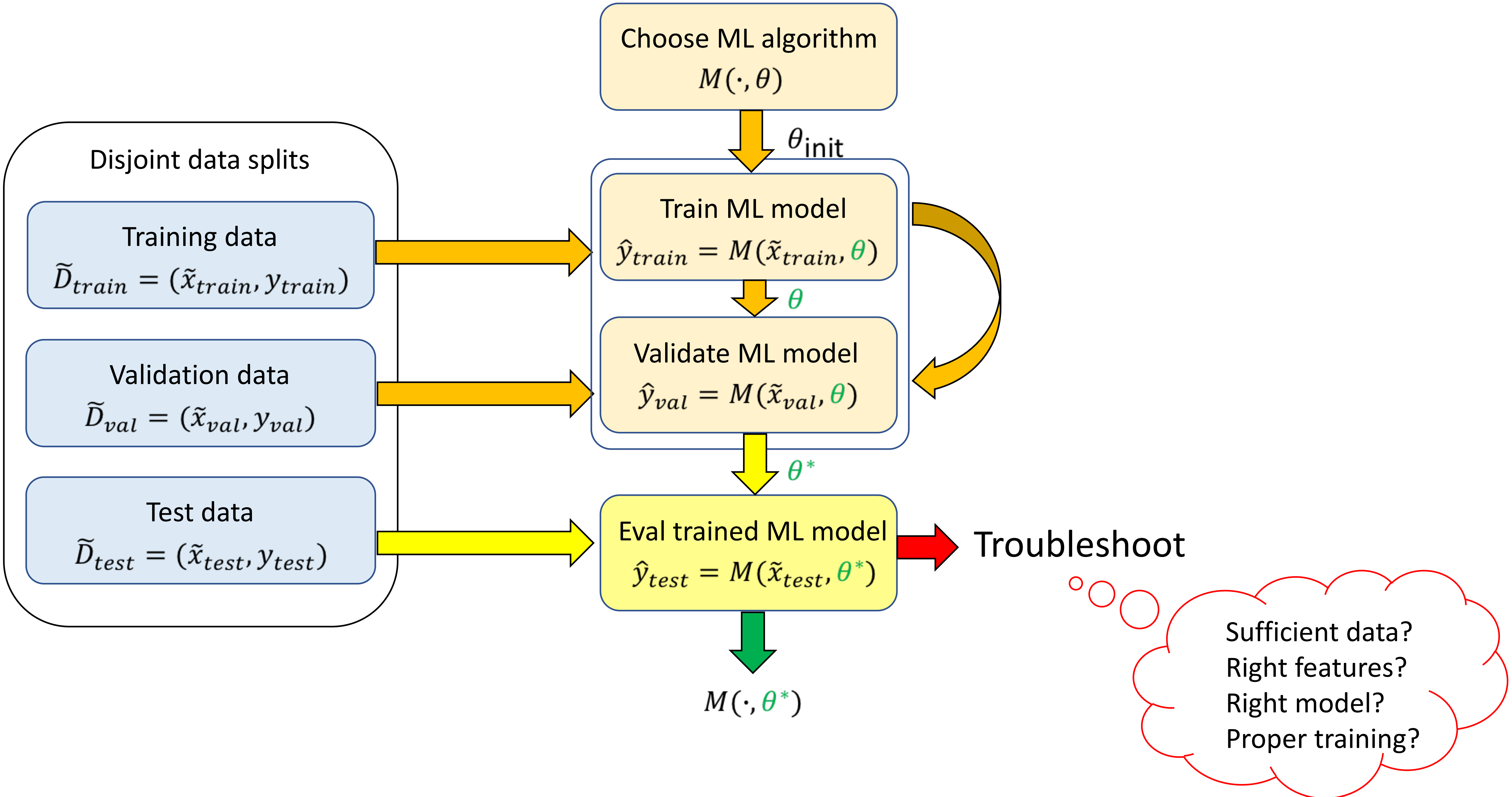
Disjoint data splits
Training data
$\widetilde{D}_{train} = (\tilde{x}_{train}, y_{train})$
Validation data
$\widetilde{D}_{val} = (\tilde{x}_{val}, y_{val})$
Test data
$\widetilde{D}_{test} = (\tilde{x}_{test}, y_{test})$
Choose ML algorithm
$M(\cdot, \theta)$
θ_{init}
Train ML model
$\hat{y}_{train} = M(\tilde{x}_{train}, \theta)$
θ
Validate ML model
$\hat{y}_{val} = M(\tilde{x}_{val}, \theta)$
Training set
Validation set
Underfitting
Optimum
Loss
θ_{ur}
θ^*
$\hat{y}$
x



Disjoint data splits
Training data
$\widetilde{D}_{train} = (\tilde{x}_{train}, y_{train})$
Validation data
$\widetilde{D}_{val} = (\tilde{x}_{val}, y_{val})$
Test data
$\widetilde{D}_{test} = (\tilde{x}_{test}, y_{test})$
Choose ML algorithm
$M(\cdot, \theta)$
θ_{init}
Train ML model
$\hat{y}_{train} = M(\tilde{x}_{train}, \theta)$
θ
Validate ML model
$\hat{y}_{val} = M(\tilde{x}_{val}, \theta)$
Training set
Validation set
Underfitting
Optimum
Overfitting
Loss
θ_{ur}
θ^*
θ_{ov}
$\hat{y}$
x

The ML workflow

1. **Define Appropriately the Problem**
2. **Collect Data**
3. **Choose a Measure of Success:**
4. **Preparing The Data**
5. **Splitting Data Into Subsets**
6. **Training the Model**
7. **Test the the Model**



Choose ML algorithm
$M(\cdot, \theta)$
θ_{init}
Disjoint data splits
Training data
$\widetilde{D}_{train} = (\tilde{x}_{train}, y_{train})$
Validation data
$\widetilde{D}_{val} = (\tilde{x}_{val}, y_{val})$
Test data
$\widetilde{D}_{test} = (\tilde{x}_{test}, y_{test})$
Train ML model
$\hat{y}_{train} = M(\tilde{x}_{train}, \theta)$
θ
Validate ML model
$\hat{y}_{val} = M(\tilde{x}_{val}, \theta)$
θ^*
Eval trained ML model
$\hat{y}_{test} = M(\tilde{x}_{test}, \theta^*)$
Troubleshoot
Sufficient data?
Right features?
Right model?
Proper training?
$M(\cdot, \theta^*)$

Evaluate your Machine Learning Algorithm

Evaluating your machine learning algorithm is an essential part of any project.

Classification Accuracy
Logarithmic Loss
Confusion Matrix
Area under Curve
F1 Score
Mean Absolute Error
Mean Squared Error

https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

The ML workflow

1. **Define Appropriately the Problem**
2. **Collect Data**
3. **Choose a Measure of Success:**
4. **Preparing The Data**
5. **Splitting Data Into Subsets**
6. **Training the Model**
7. **Test the the Model**
8. **Tune Hyperparameters**

https://medium.com/criteo-labs/hyper-parameter-optimization-algorithms-2fe447525903

Introduction to Machine Learning using Scikit-Learn

Scikit-learn is probably the most useful library for machine **learning** in Python. The sklearn library contains a lot of efficient tools for machine **learning** and statistical modeling including classification, regression, clustering and dimensionality reduction

Tutorials
https://scikit-learn.org/stable/
https://scikit-learn.org/stable/tutorial/basic/tutorial.html
https://www.guru99.com/scikit-learn-tutorial.html

ML Projects for beginners
https://github.com/jakevdp/sklearn_tutorial
https://www.guru99.com/scikit-learn-tutorial.html
https://elitedatascience.com/machine-learning-projects-for-beginners