UiO : Universitetet i Oslo

# Machine Learning in Computational Biology: Introduction to ML

IN-BIOS5000/IN-BIOS9000

Milena Pavlović
Biomedical Informatics Research Group
Department of Informatics

milenpa@student.matnat.uio.no

# Machine learning in computational biology - outline

- **Introduction to machine learning:**
  - **What is machine learning, types of problems, assumptions, workflow, generalization**
- Machine learning models and algorithms:
  - supervised models (logistic and linear regression, kNN, neural networks), unsupervised models (dimensionality reduction, clustering)
- Data representation:
  - Considerations and examples, one-hot encoding, feature engineering, representation learning
- Model comparison and uncertainty:
  - Model assessment, model selection, uncertainty, cross-validation
- Transparency and reproducibility

# What is machine learning?

“Machine learning refers to extracting patterns from raw data.”

# What is machine learning?

“Machine learning refers to extracting patterns from raw data.”

“Machine learning is essentially a form of applied statistics with increased emphasis in the use of computers to statistically estimate complicated functions and a decreased emphasis in proving confidence intervals around these functions.”

Goodfellow et al. 2016

# What is machine learning?

“Machine learning refers to extracting patterns from raw data.”

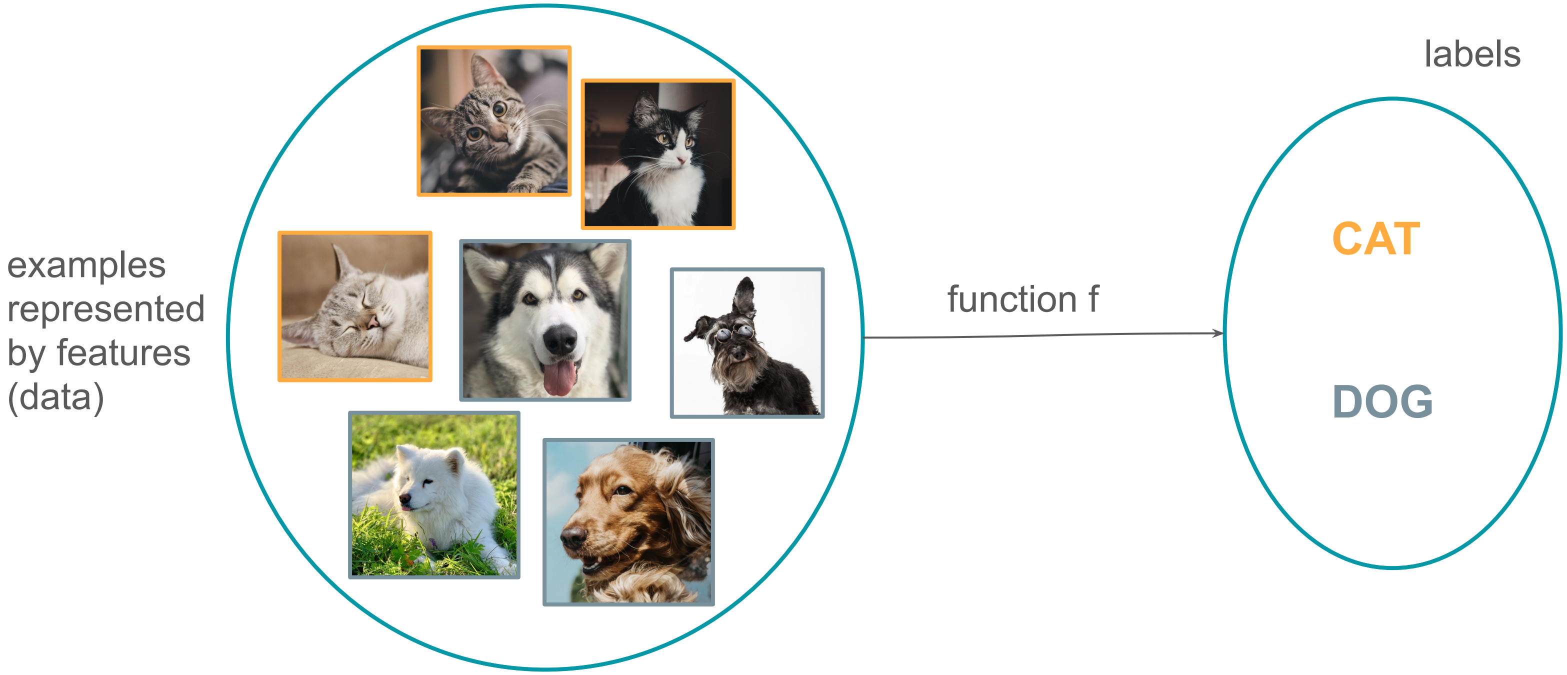
“Machine learning is essentially a form of applied statistics with increased emphasis in the use of computers to statistically estimate complicated functions and a decreased emphasis in proving confidence intervals around these functions.”

Goodfellow et al. 2016

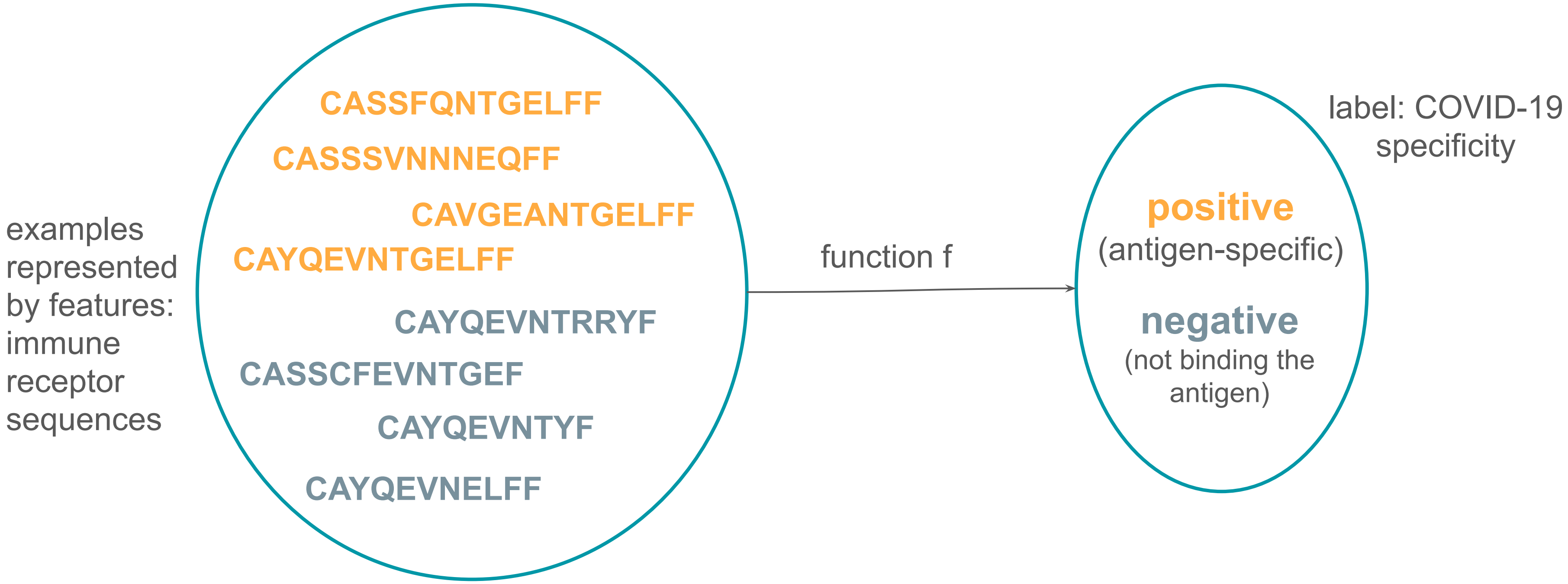
“A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”

Mitchell 1997

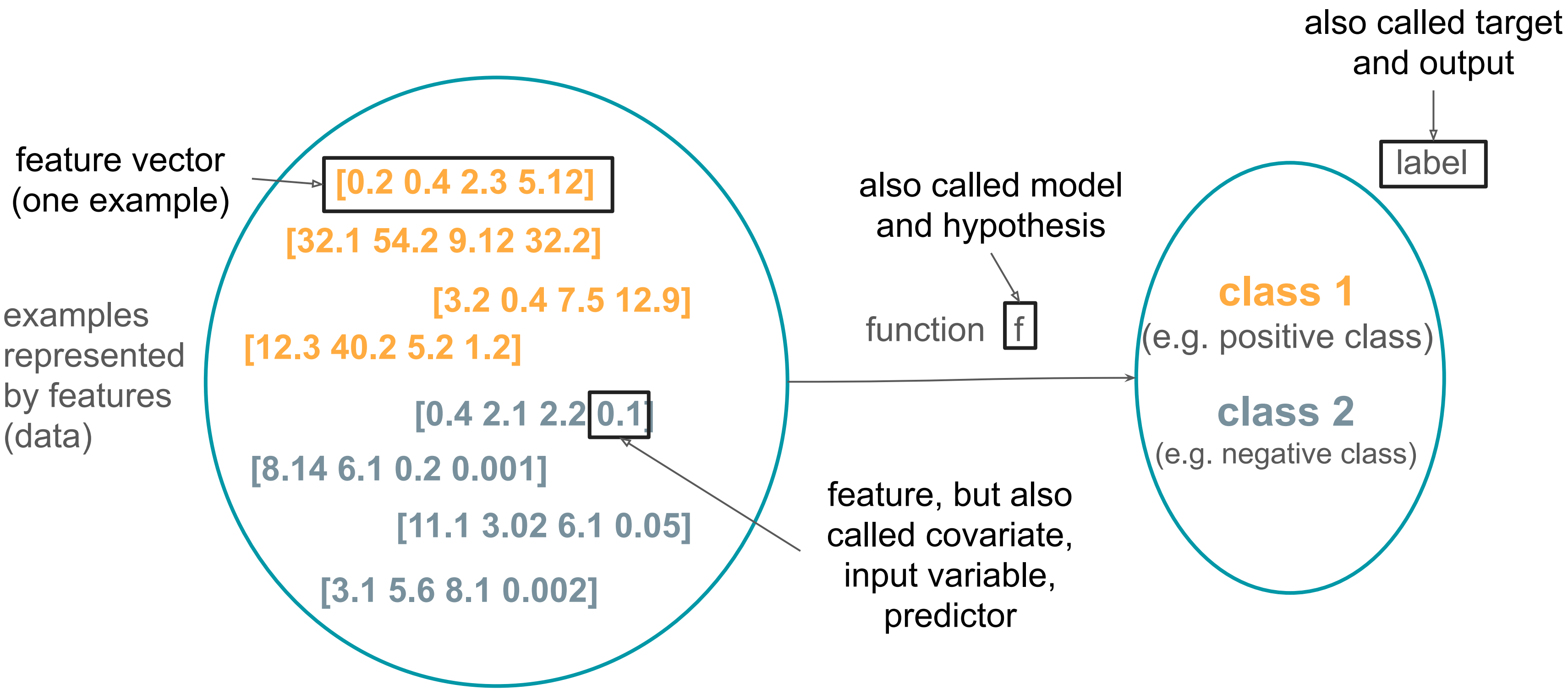
# Machine learning as a function approximation task



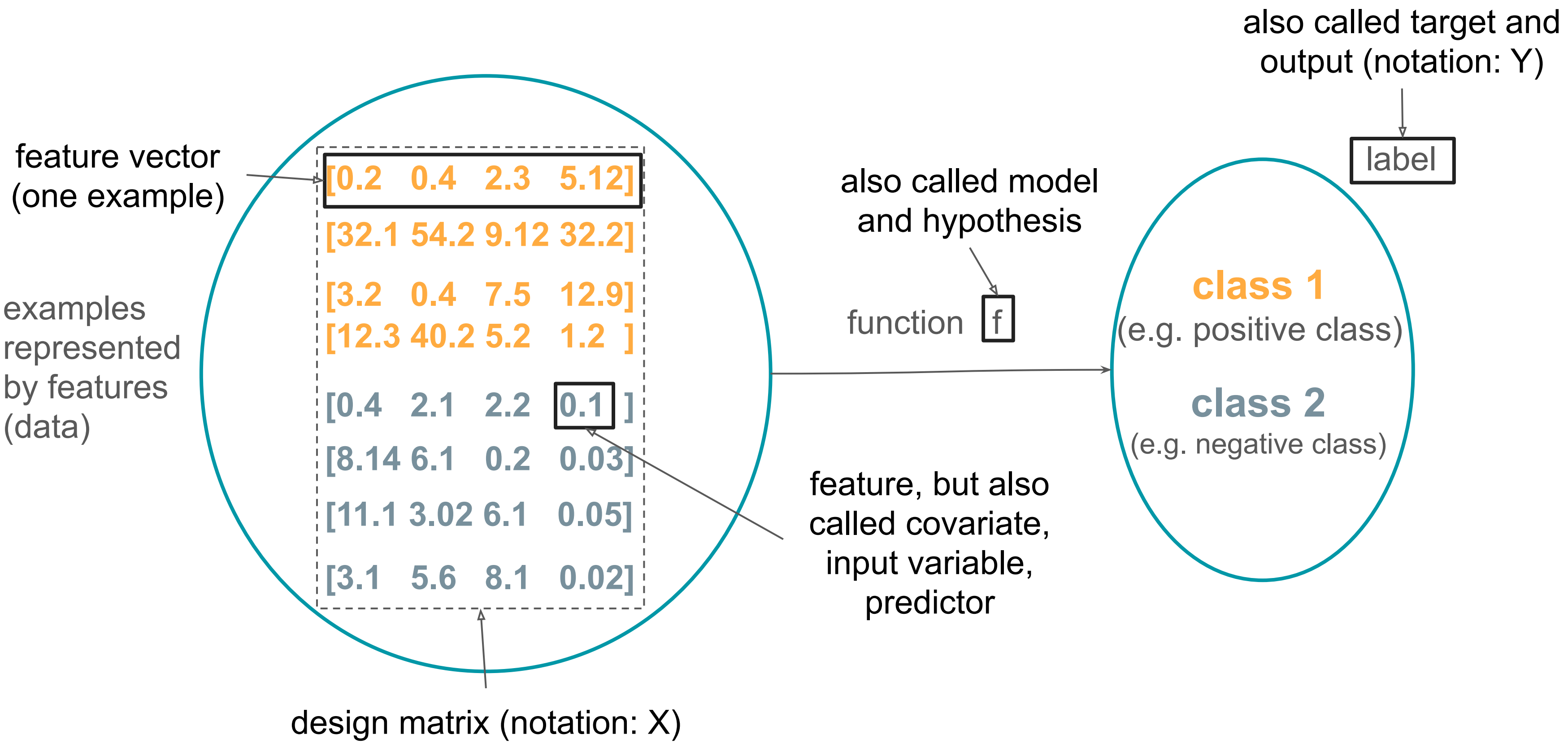
# Machine learning as a function approximation task



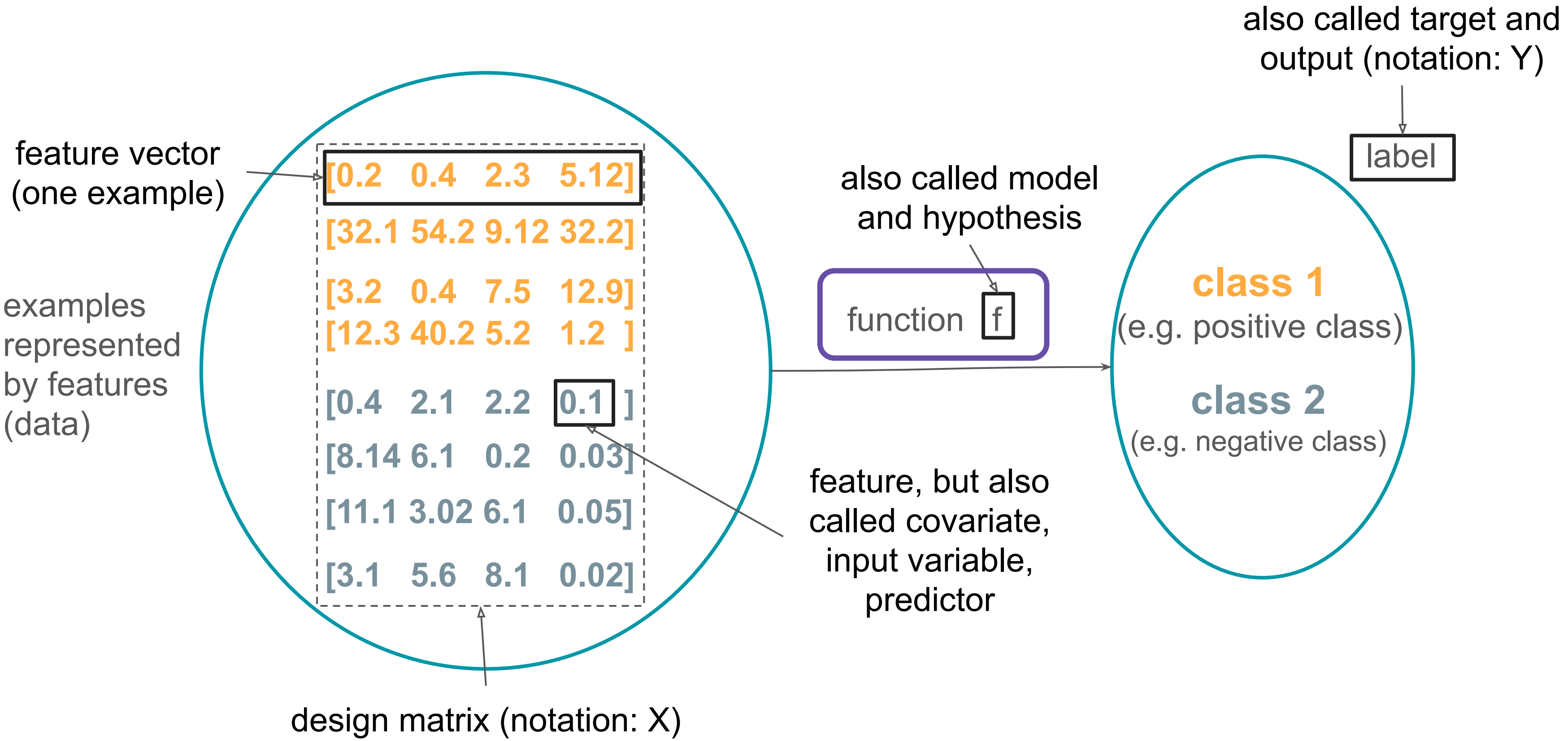
# Machine learning as a function approximation task



# Machine learning as a function approximation task

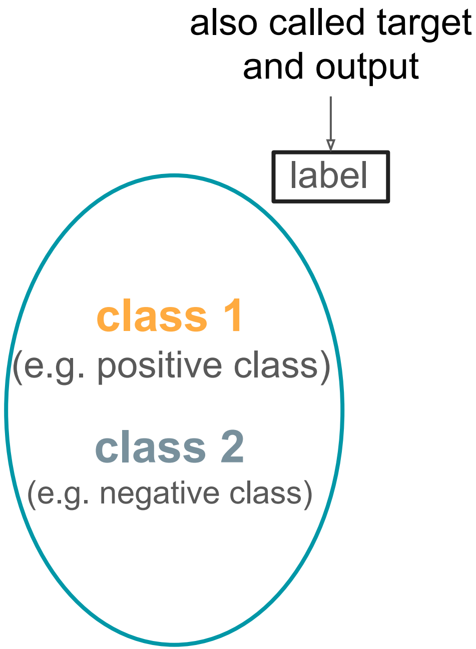


# Machine learning as a function approximation task



# Types of problems in machine learning: what does the function f do

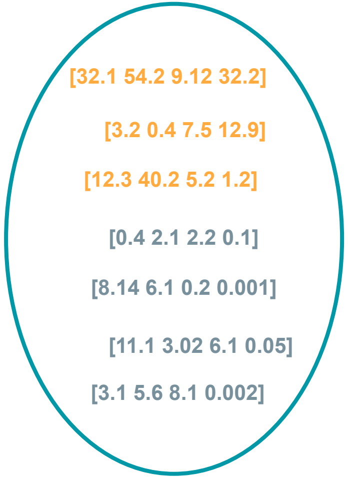
1. **Supervised**: for each example we know the label
    a. Label can be a discrete value - a class (e.g., a receptor is antigen-specific or not, the picture contains a dog or a cat): **classification** or
    b. a continuous value (e.g., binding affinity, house price): **regression**



# Types of problems in machine learning: what does the function f do

1. **Supervised**: **classification** and **regression**
2. **Unsupervised**:
    a. the data we have has a lot of features, and we want to see if there is a structure in the data
    b. there is no explicit label
    c. example: there is a set of cells and we want to see if we can group them and see if there are new groups which could indicate new cell types

no label, just data



# Types of problems in machine learning: what does the function f do

1. **Supervised**: **classification** and **regression**
2. **Unsupervised**
3. **Reinforcement learning**:
    a. dataset is not fixed, the program interacts with environment
    b. used when choosing a sequence of actions: we don't know the label - don't know the optimal sequence of actions, but we know how good an action is
    c. example: discover optimal dosing policy for a medication

# Types of problems in machine learning: what does the function f do

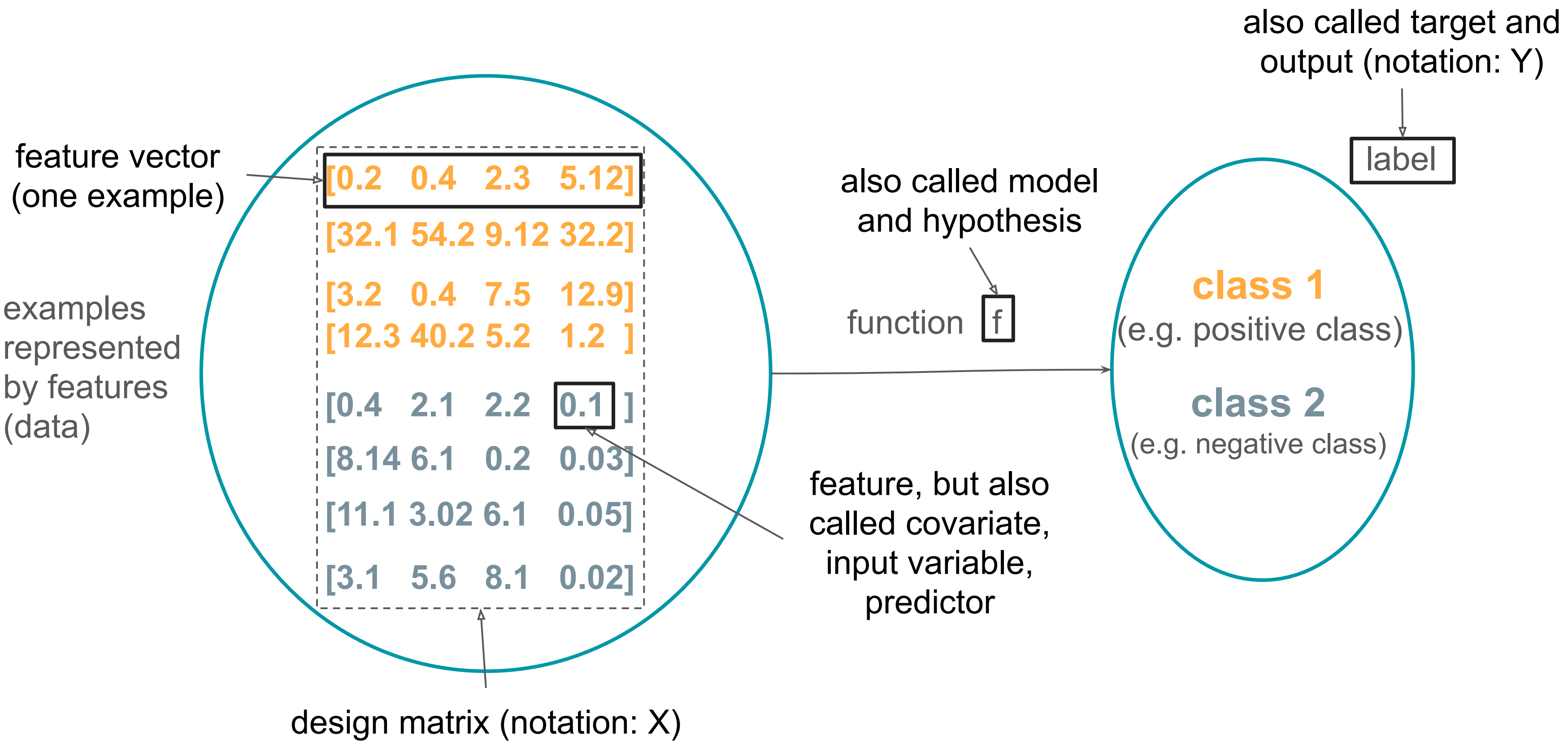
1. **Supervised**: **classification** and **regression**
2. **Unsupervised**
3. **Reinforcement learning**

# Types of problems in machine learning: what does the function f do

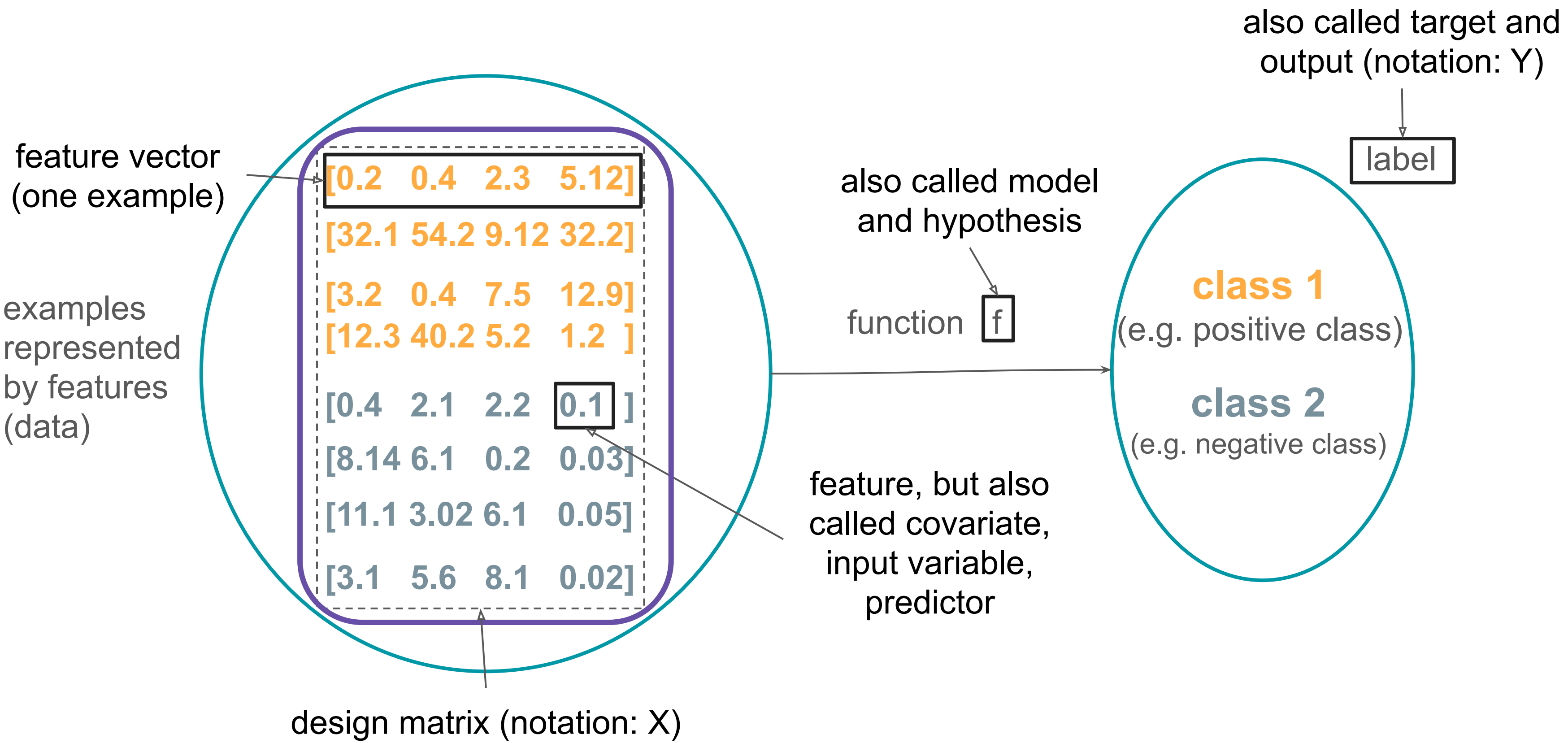
1. **Supervised**: **classification** and **regression**
2. **Unsupervised**
3. **Reinforcement learning**

And we will talk more about the specific forms the function f can take and how we choose the right (or the least wrong) one

# Machine learning as a function approximation task



# Machine learning as a function approximation task



# What do we assume about the data?

- Data generation process produces the data
  (data generation process results in a probability distribution $p_{data}$)
- We assume:
  - Examples in each dataset are independent of each other
  - When we want to use the machine learning model on some new data to predict a label: these new data come from the same data generation process (same probability distribution)

# What do we assume about the data?

- Data generation process produces the data (data generation process results in a probability distribution $p_{data}$)
- We assume:
  - Examples in each dataset are independent of each other
  - When we want to use the machine learning model on some new data to predict a label: these new data come from the same data generation process (same probability distribution)

# What do we assume about the data?

- ❑ Data generation process produces the data
  (data generation process results in a probability distribution $p_{data}$)
- ❑ We assume:
  - ❑ Examples in each dataset are independent of each other
  - ❑ When we want to use the machine learning model on some new data to predict a label: these new data come from the same data generation process (same probability distribution)

**i.i.d. assumption**

examples are independent and identically distributed

# What do we assume about the data?

- ❏ Data generation process produces the data
  (data generation process results in a probability distribution $p_{data}$)
- ❏ We assume:
  - ❏ Examples in each dataset are independent of each other
  - ❏ When we want to use the machine learning model on some new data to predict a label: these new data come from the same data generation process (same probability distribution)
- ❏ With these assumptions satisfied (or approximately satisfied), we choose the data representation and estimate the function

**i.i.d. assumption**

↓

examples are independent and identically distributed

# What do we assume about the data?

- ❑ Data generation process produces the data
  (data generation process results in a probability distribution $p_{data}$)
- ❑ We assume:
  - ❑ Examples in each dataset are independent of each other
  - ❑ When we want to use the machine learning model on some new data to predict a label: these new data come from the same data generation process (same probability distribution)

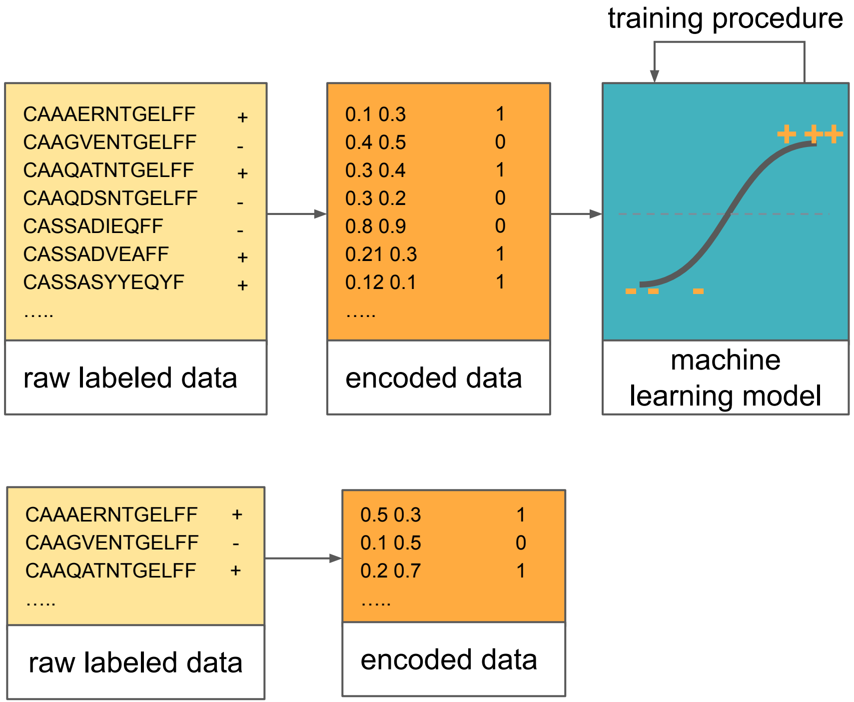
**i.i.d. assumption**

examples are independent and identically distributed

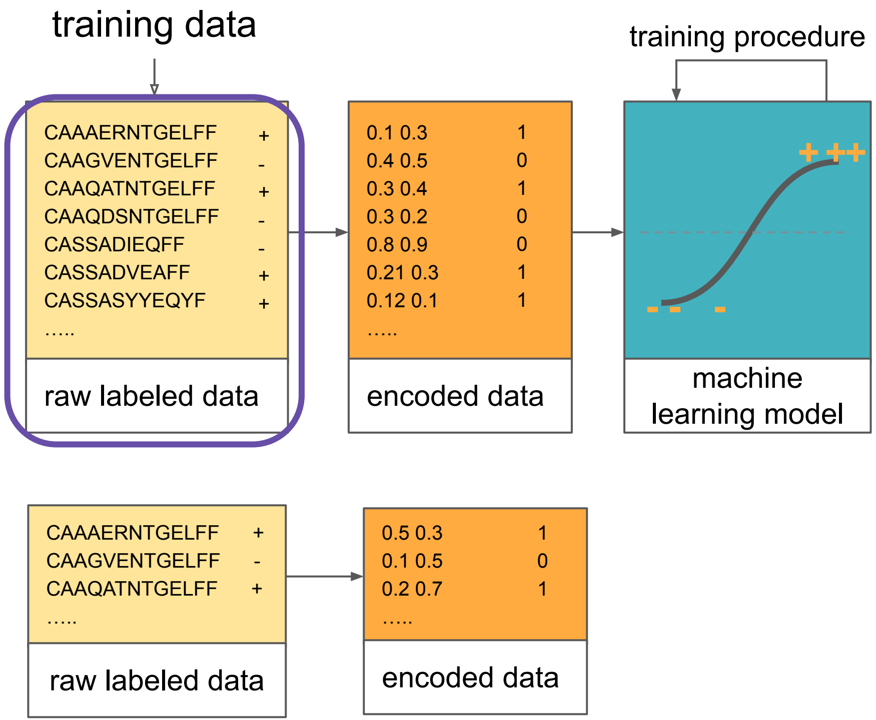
- ❑ With these assumptions satisfied (or approximately satisfied), we choose the data representation and estimate the function

And we will talk more about how to represent the data

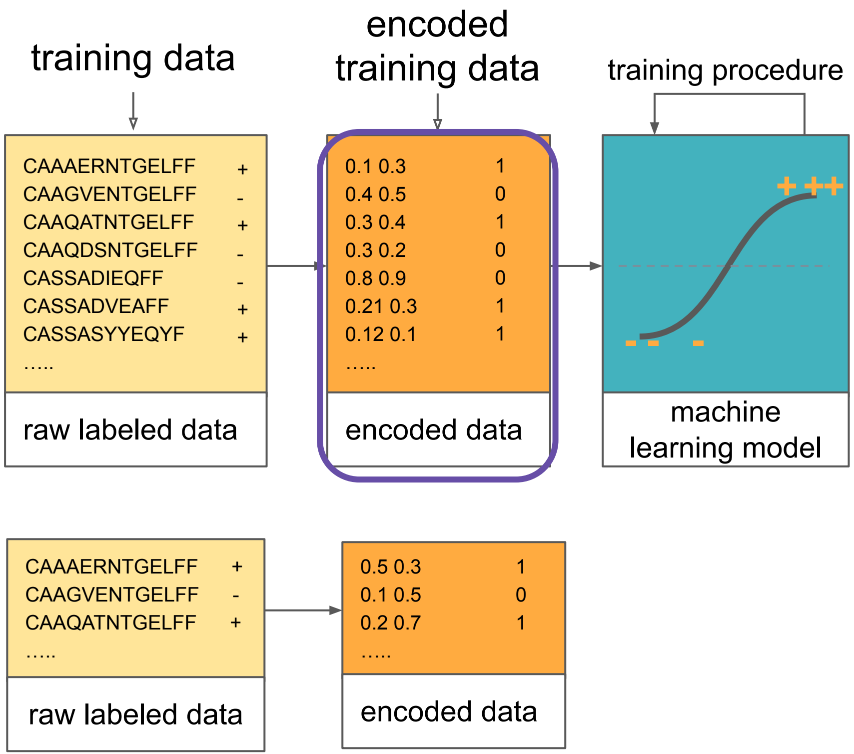
# Estimating the function (training procedure)



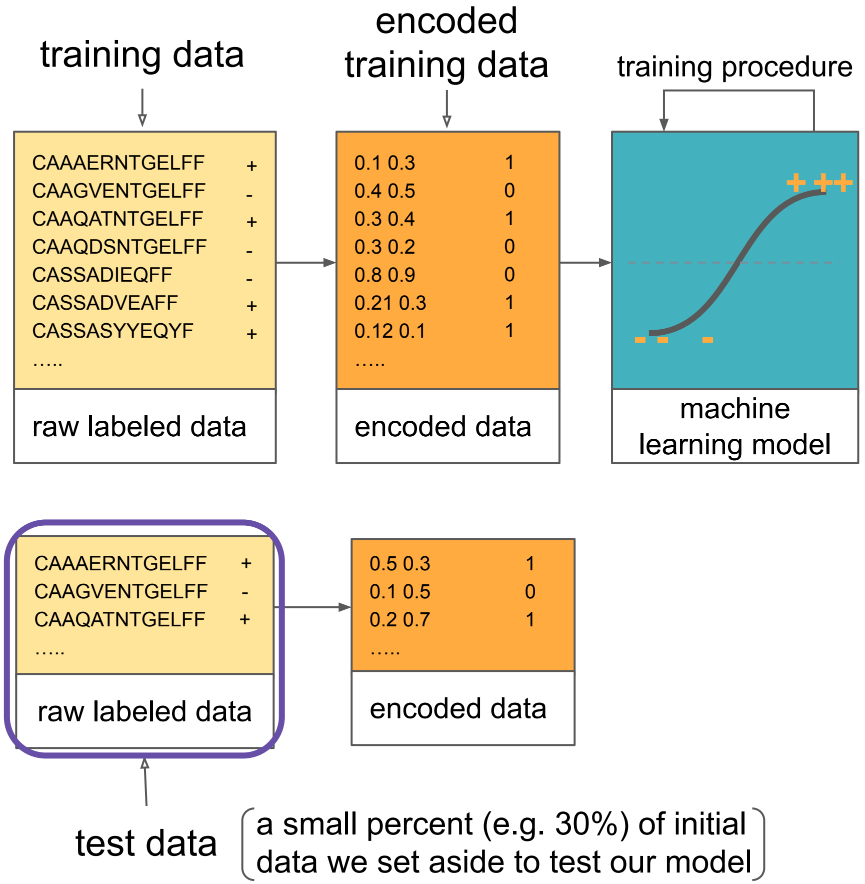
# Estimating the function (training procedure)



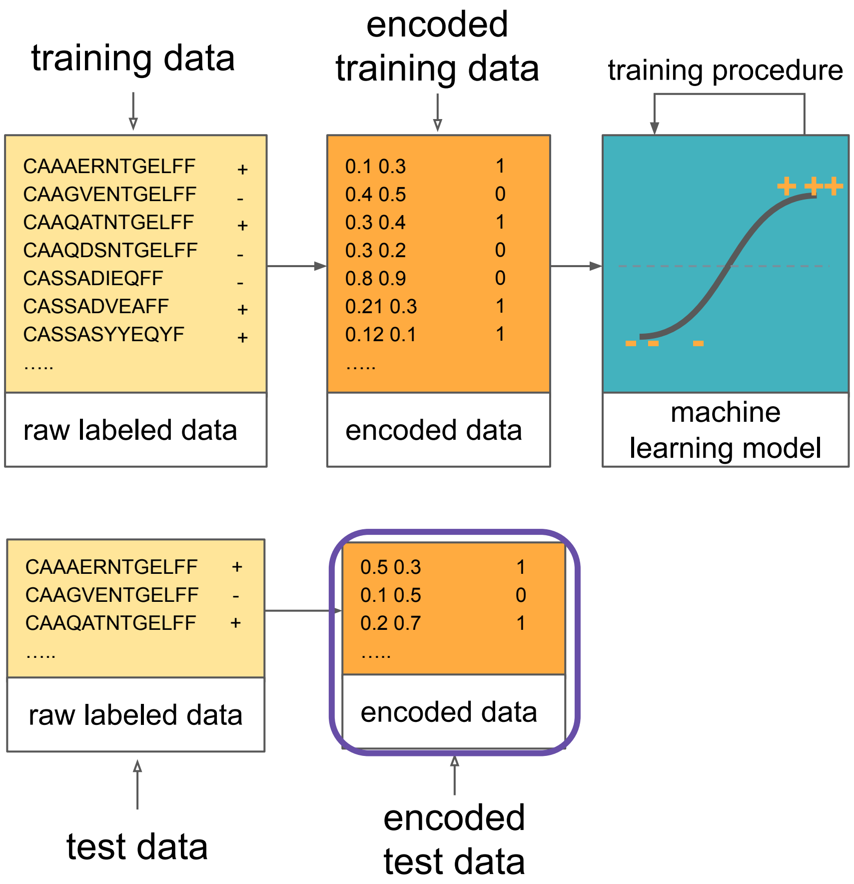
# Estimating the function (training procedure)



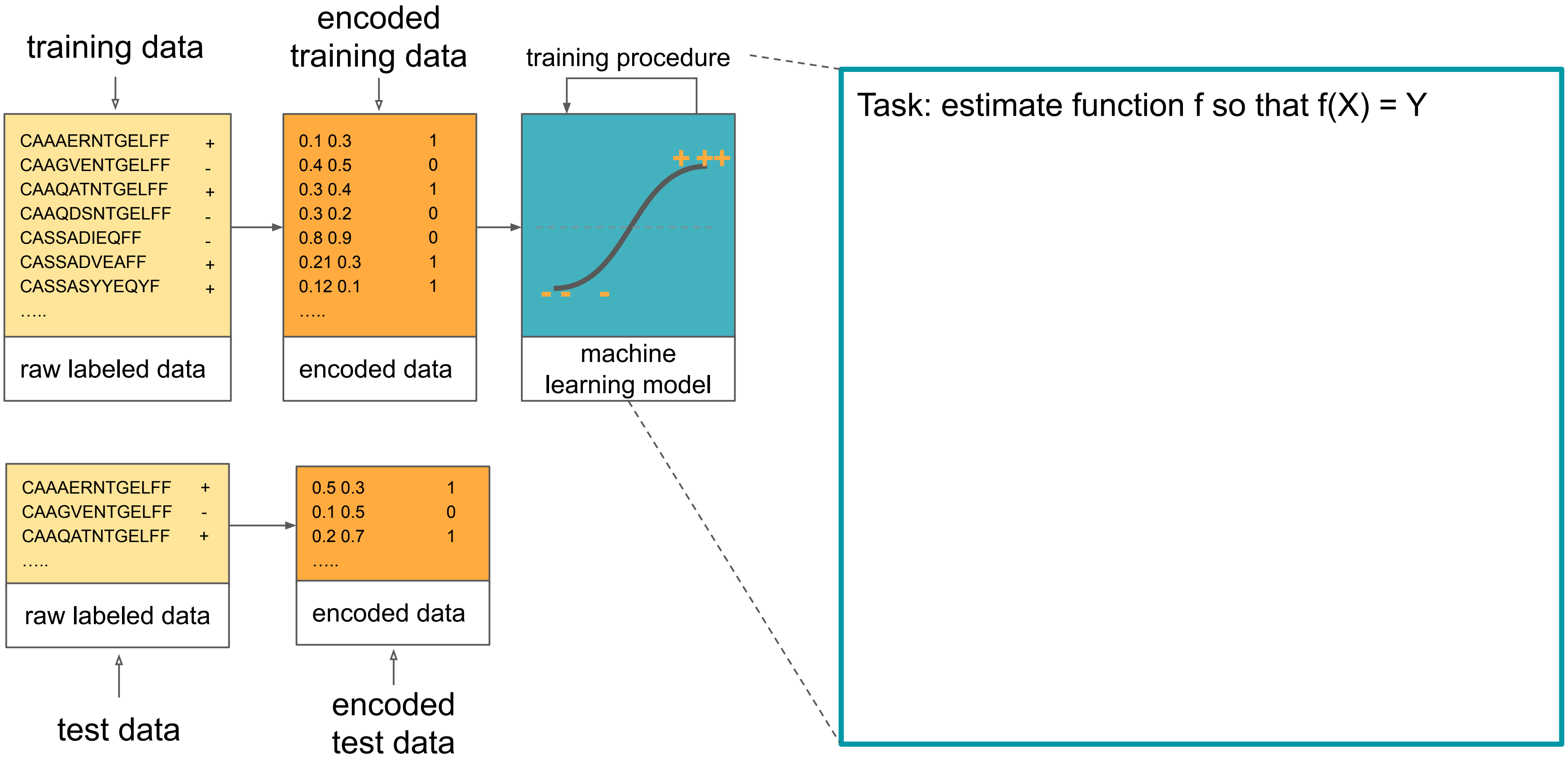
# Estimating the function (training procedure)



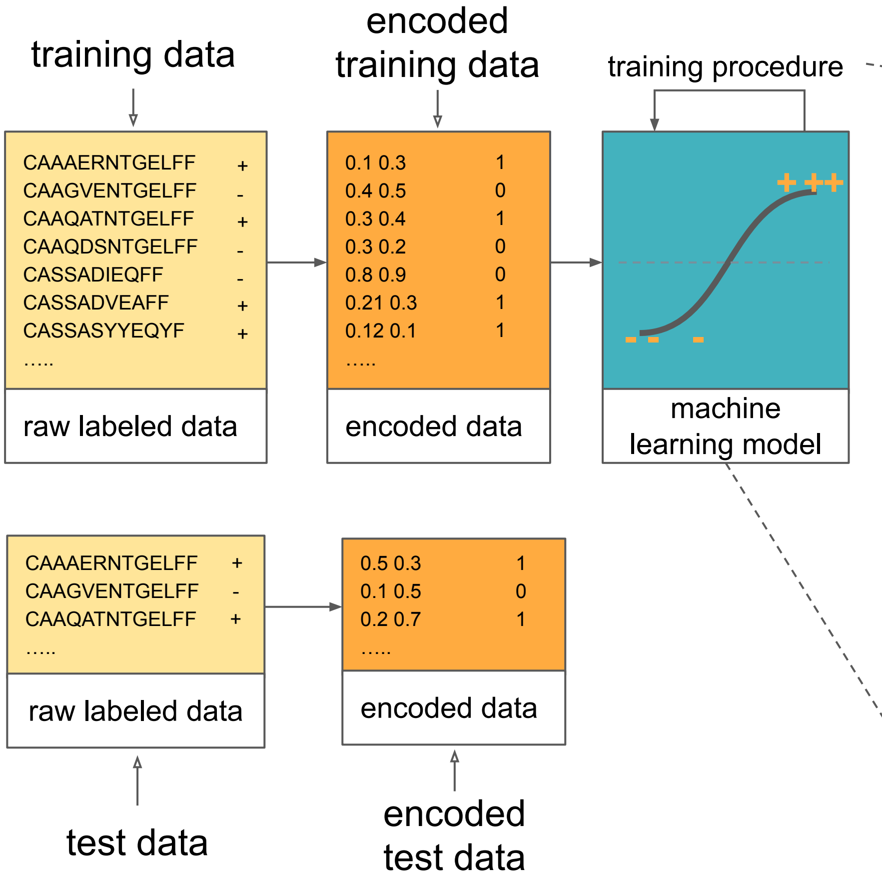
# Estimating the function (training procedure)



# Estimating the function (training procedure)



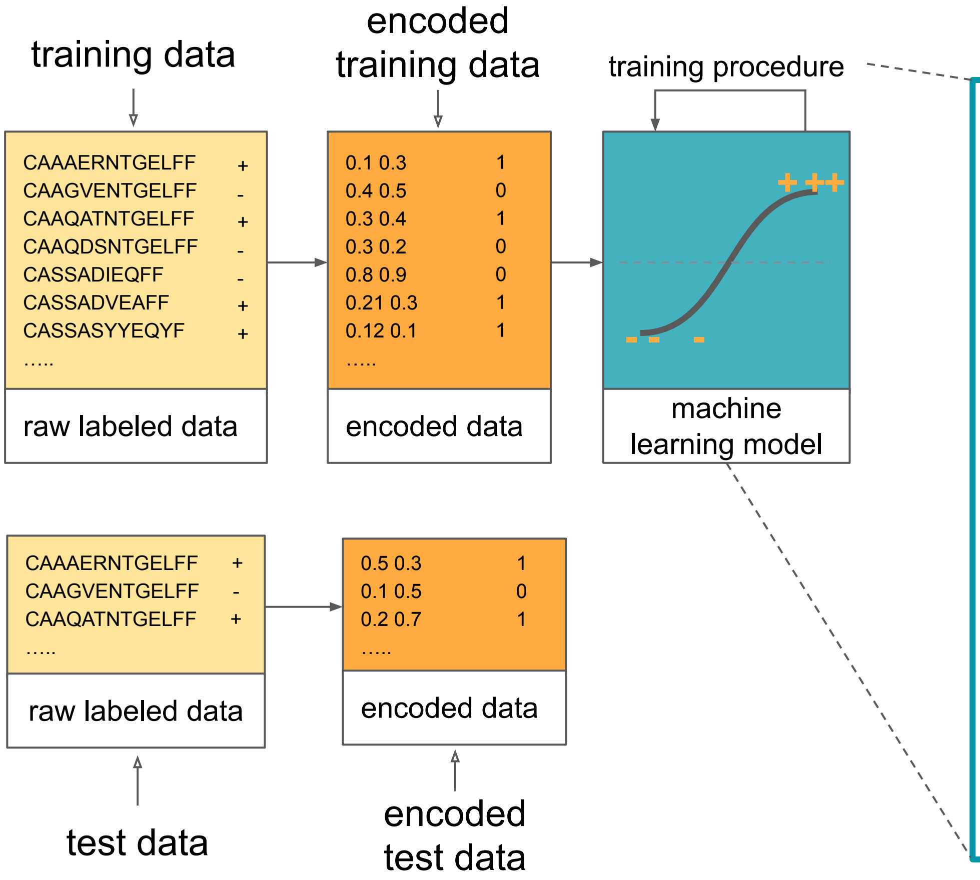
# Estimating the function (training procedure)




Task: estimate function f so that f(X) = Y

Training procedure:

# Estimating the function (training procedure)

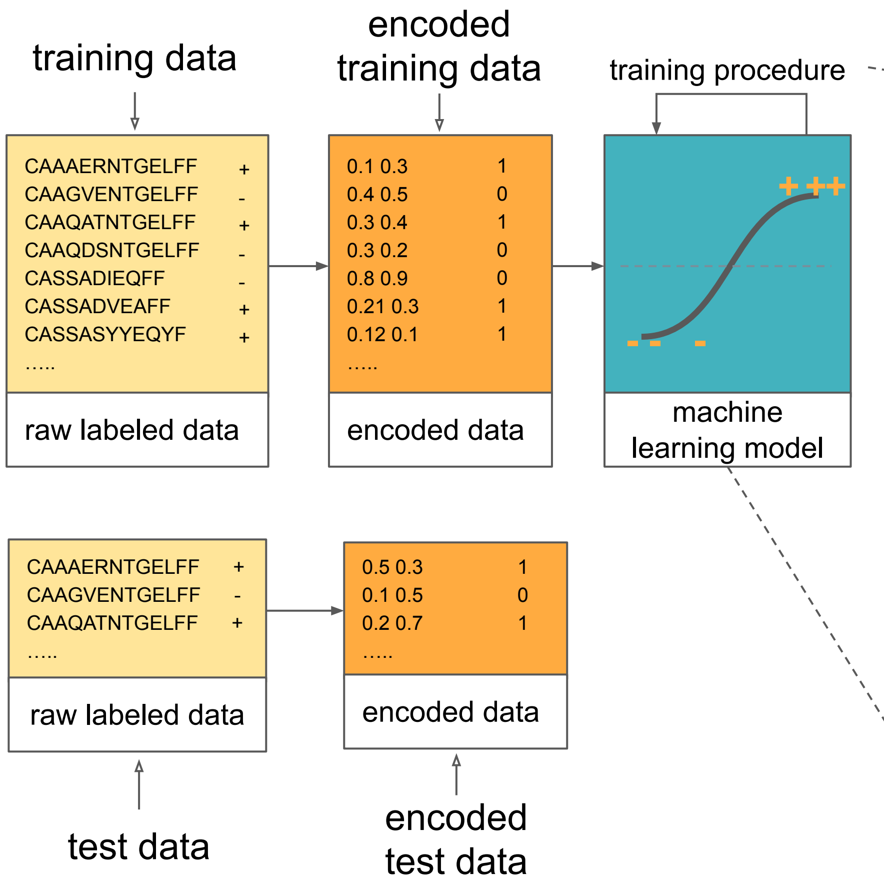



Task: estimate function f so that f(X) = Y

Training procedure:

1. Start with some function f with some parameters

# Estimating the function (training procedure)




Task: estimate function f so that f(X) = Y

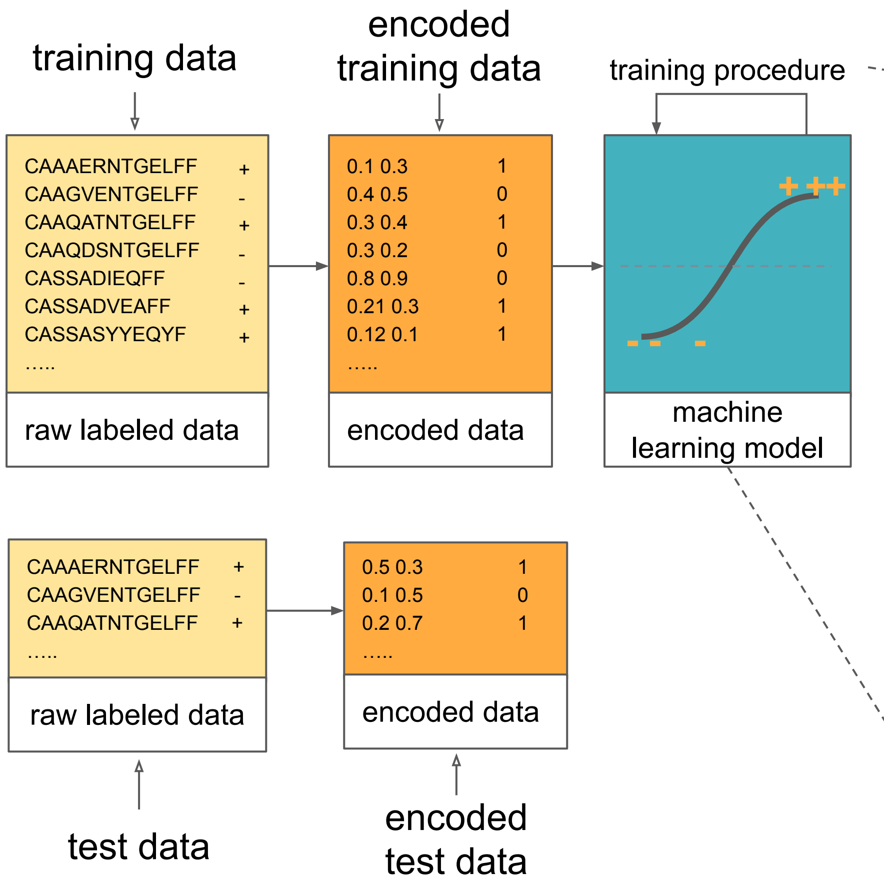
Training procedure:

1. Start with some function f with some parameters
   for example, logistic regression:

$$g(\omega x + b) = \left(1 + e^{-(\omega x + b)}\right)^{-1}$$

$$f(x) = \begin{cases} 1, & g(\omega x + b) \geq 0.5 \\ 0, & g(\omega x + b) < 0.5 \end{cases}$$

# Estimating the function (training procedure)

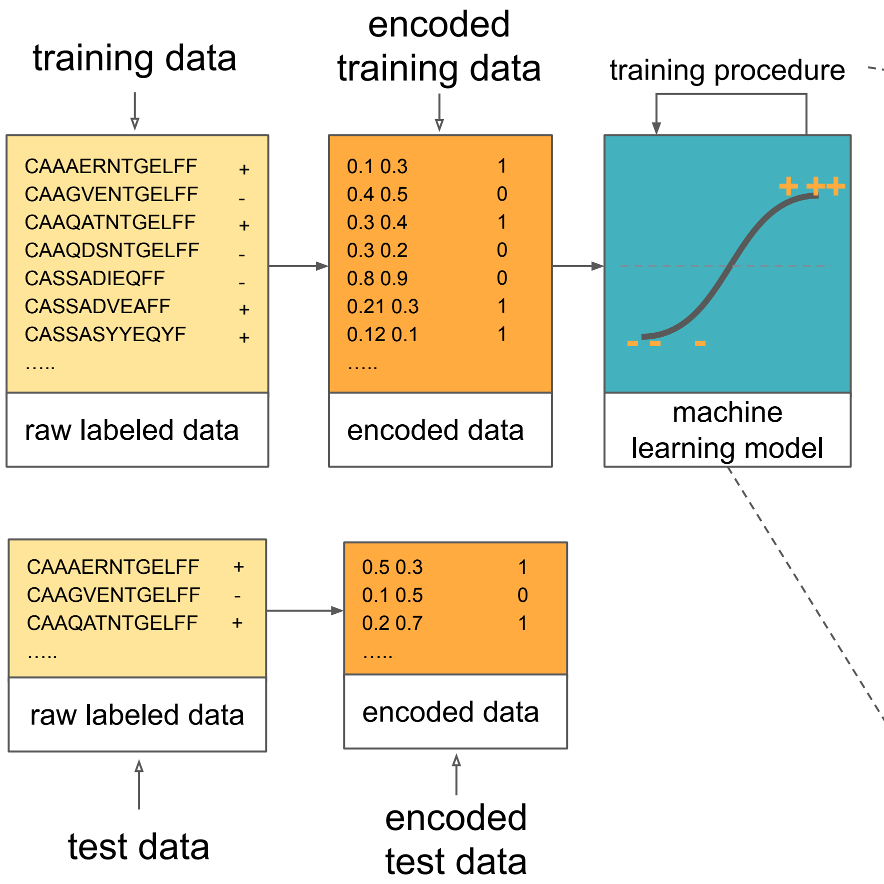



Task: estimate function f so that f(X) = Y

Training procedure:

1. Start with some function f with some parameters (e.g., logistic regression)

# Estimating the function (training procedure)

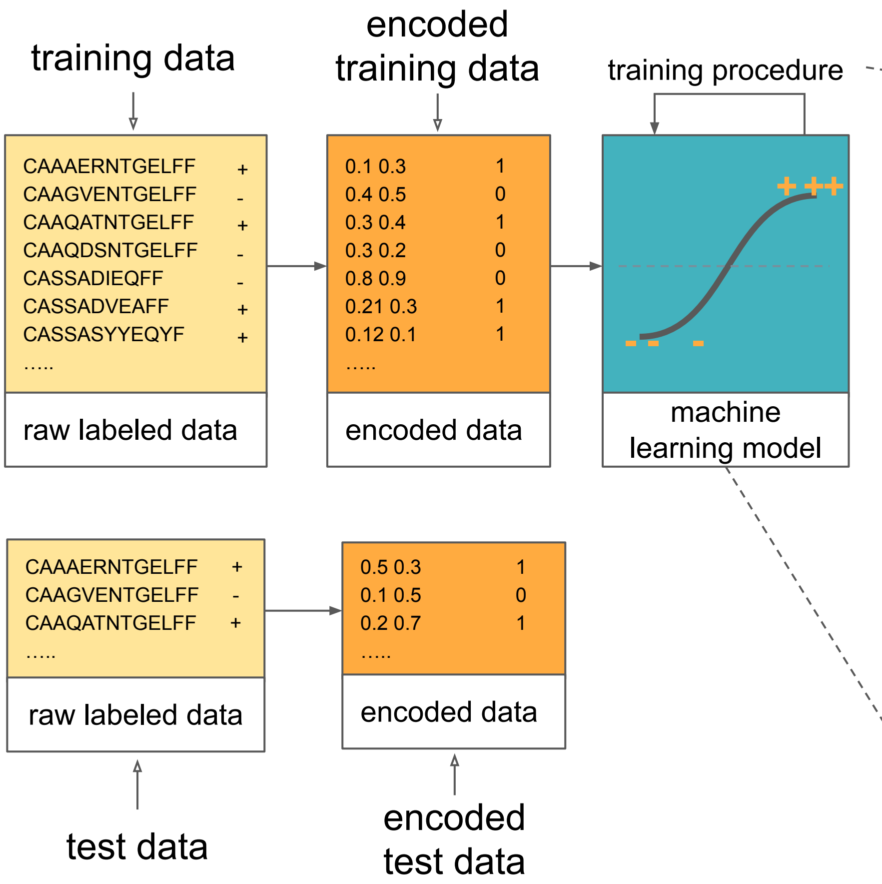



Task: estimate function f so that f(X) = Y

Training procedure:

1. Start with some function f with some parameters (e.g., logistic regression)
2. While training:

# Estimating the function (training procedure)



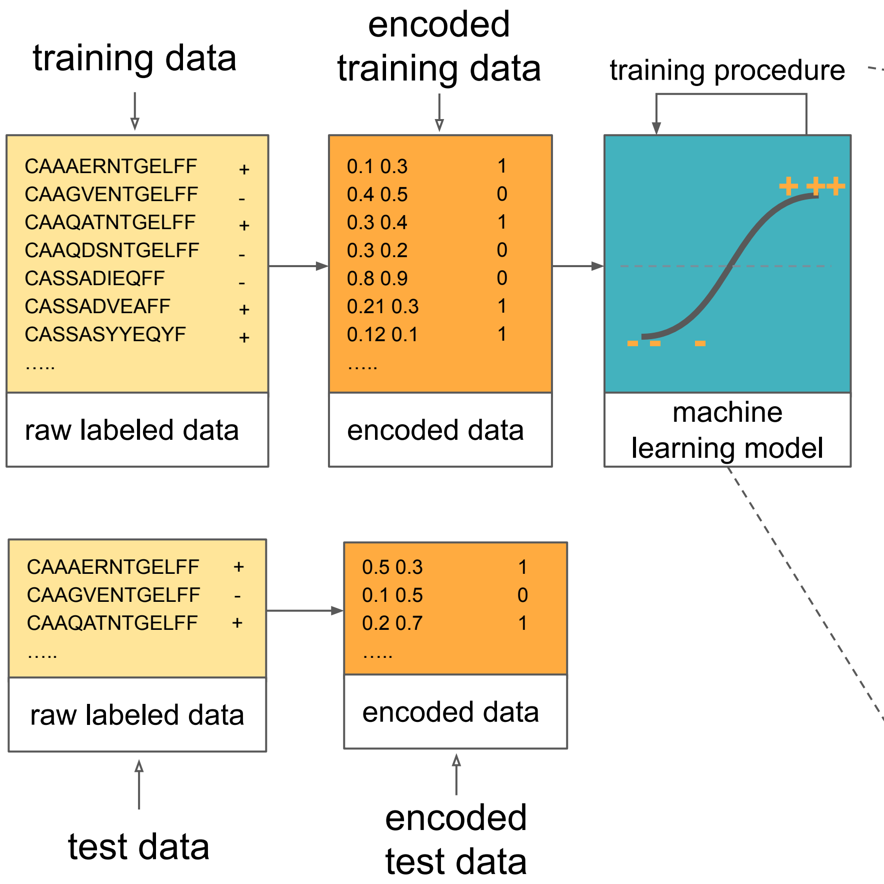

Task: estimate function f so that f(X) = Y

Training procedure:

1. Start with some function f with some parameters (e.g., logistic regression)
2. While training:

max number of iterations was not reached and predictions are not good enough

# Estimating the function (training procedure)

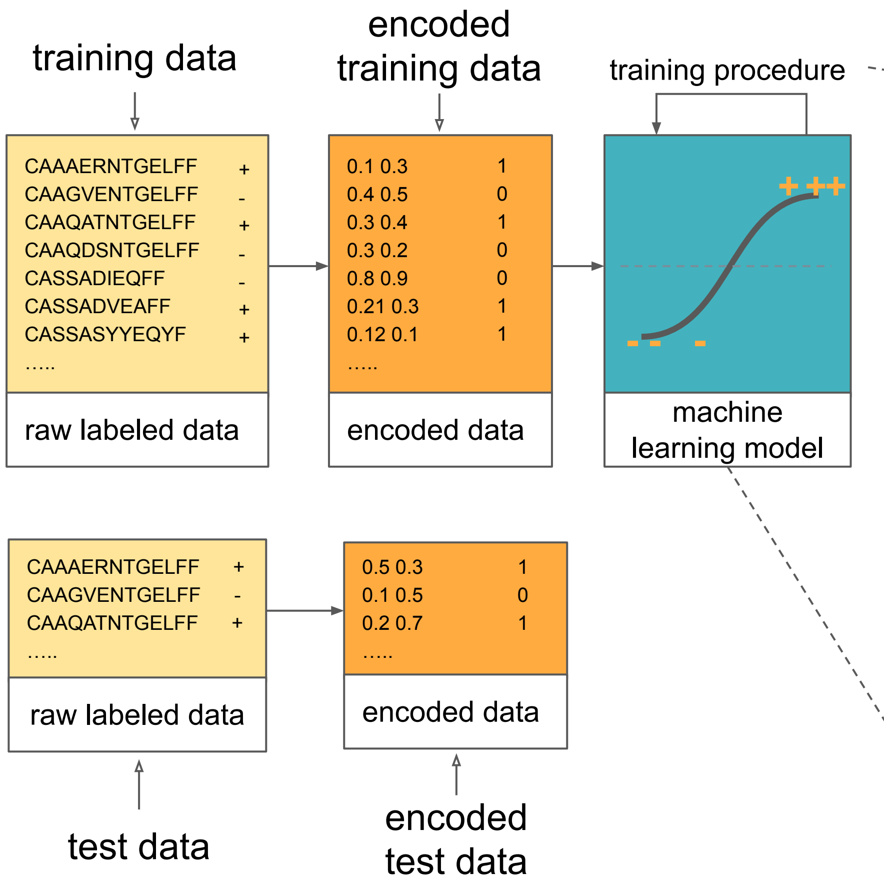



Task: estimate function f so that f(X) = Y

Training procedure:

1. Start with some function f with some parameters (e.g., logistic regression)
2. While training:

# Estimating the function (training procedure)

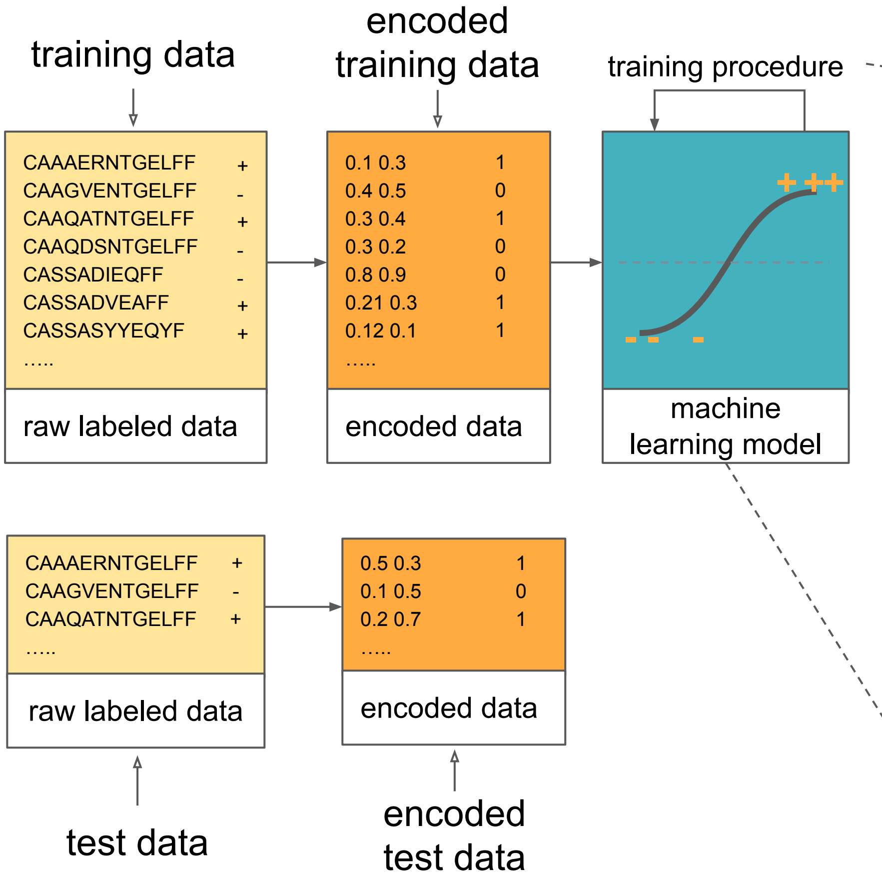



Task: estimate function f so that f(X) = Y

Training procedure:

1. Start with some function f with some parameters (e.g., logistic regression)
2. While training:
    a. Predict the label Y from the encoded training data X

# Estimating the function (training procedure)

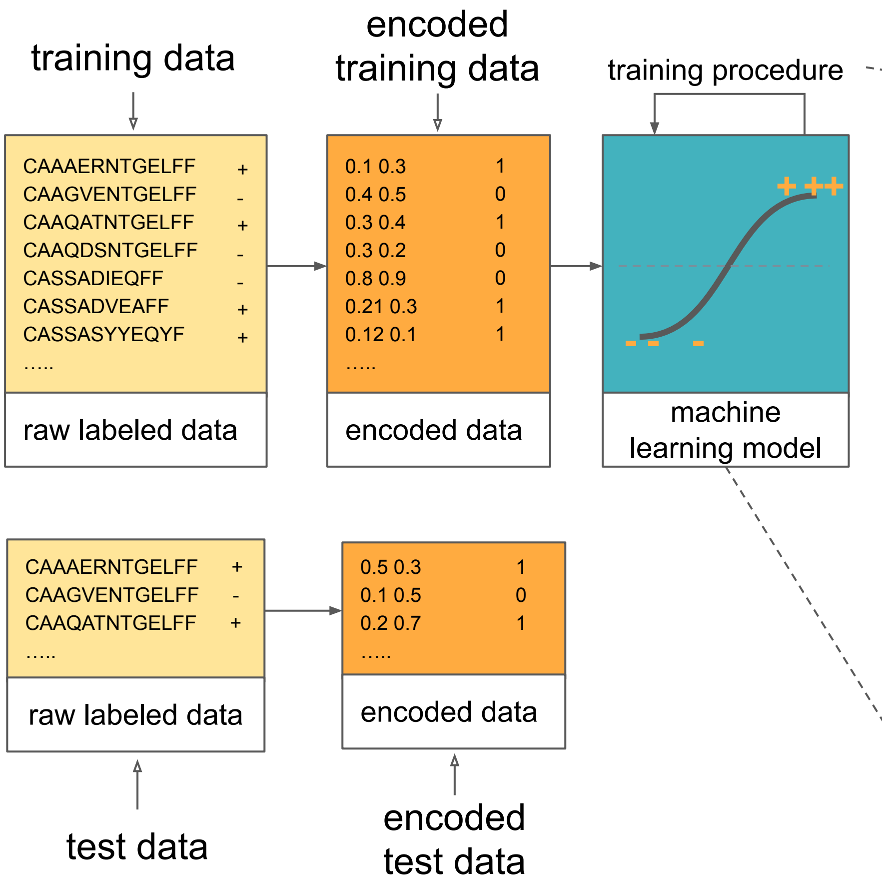



Task: estimate function f so that f(X) = Y

Training procedure:

1. Start with some function f with some parameters (e.g., logistic regression)
2. While training:
   a. Predict the label Y from the encoded training data X
   b. Compute the cost function: how much predictions deviate from the label Y

# Estimating the function (training procedure)

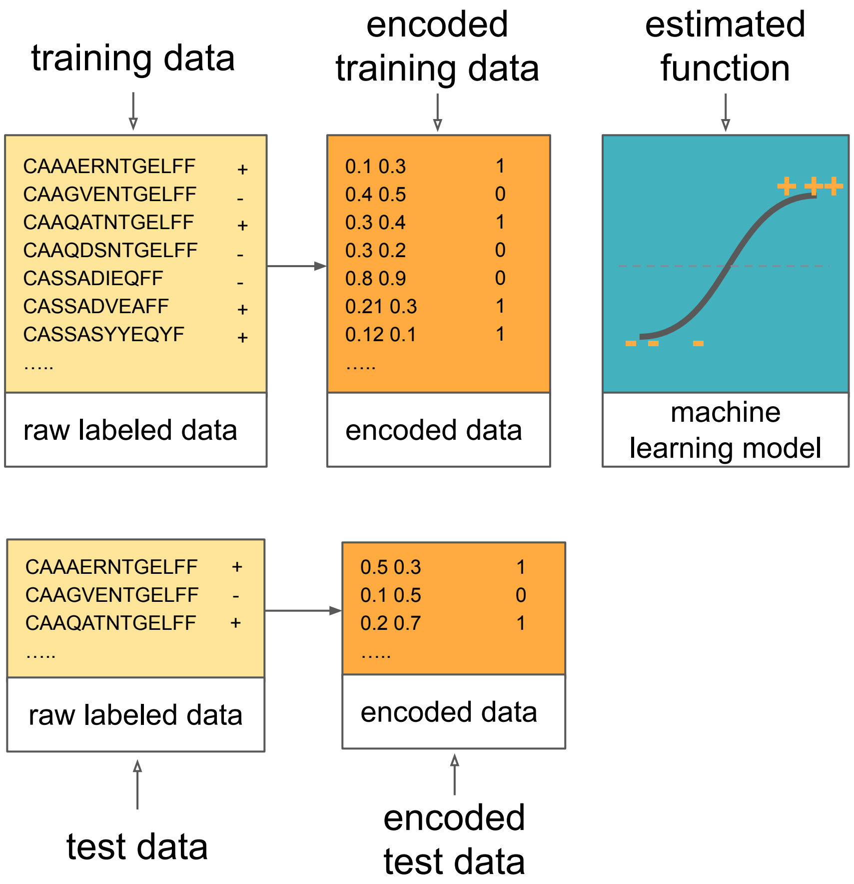



Task: estimate function f so that f(X) = Y

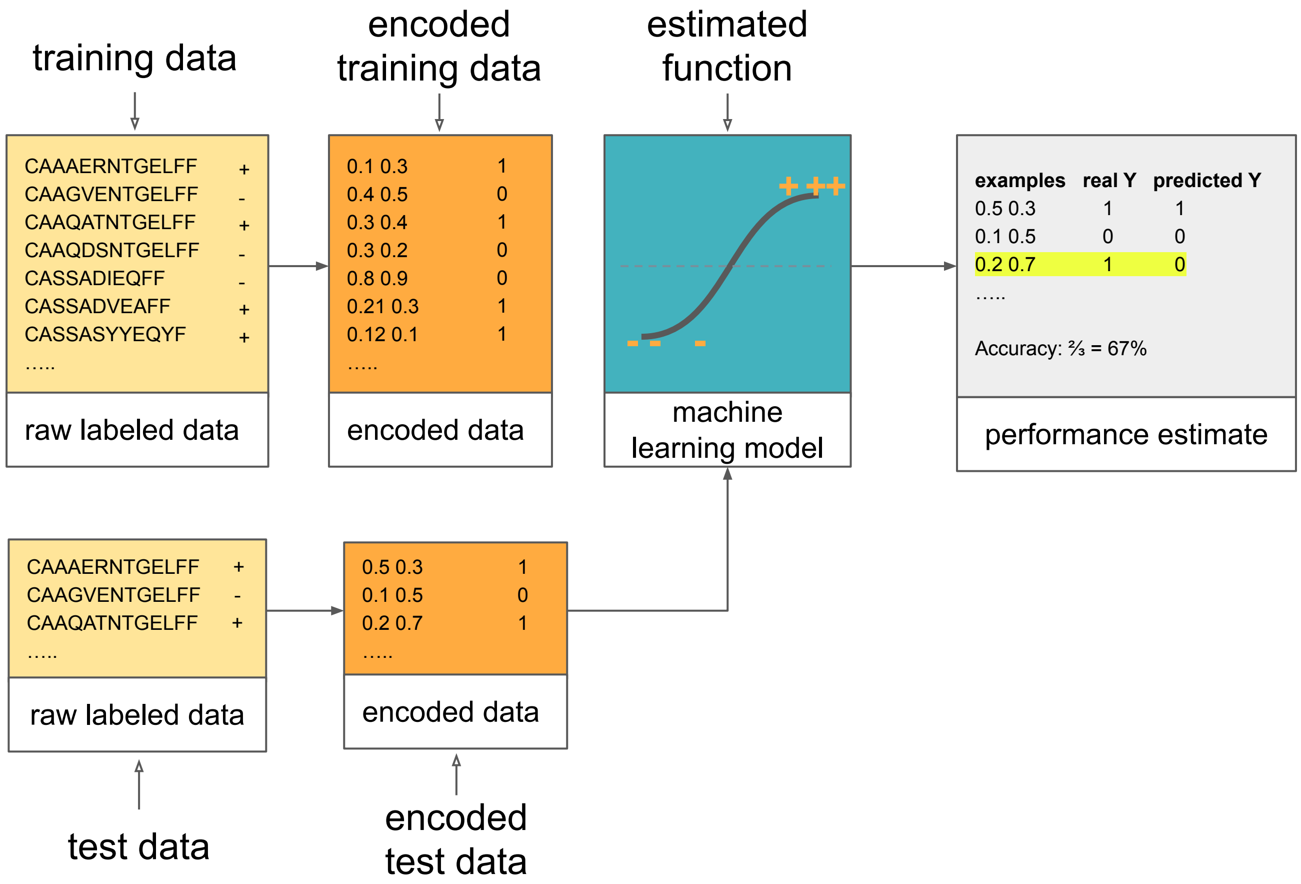
Training procedure:

1. Start with some function f with some parameters (e.g., logistic regression)
2. While training:
   a. Predict the label Y from the encoded training data X
   b. Compute the cost function: how much predictions deviate from the label Y
   c. Update the parameters of the function f to reduce the cost function so that we get better predictions

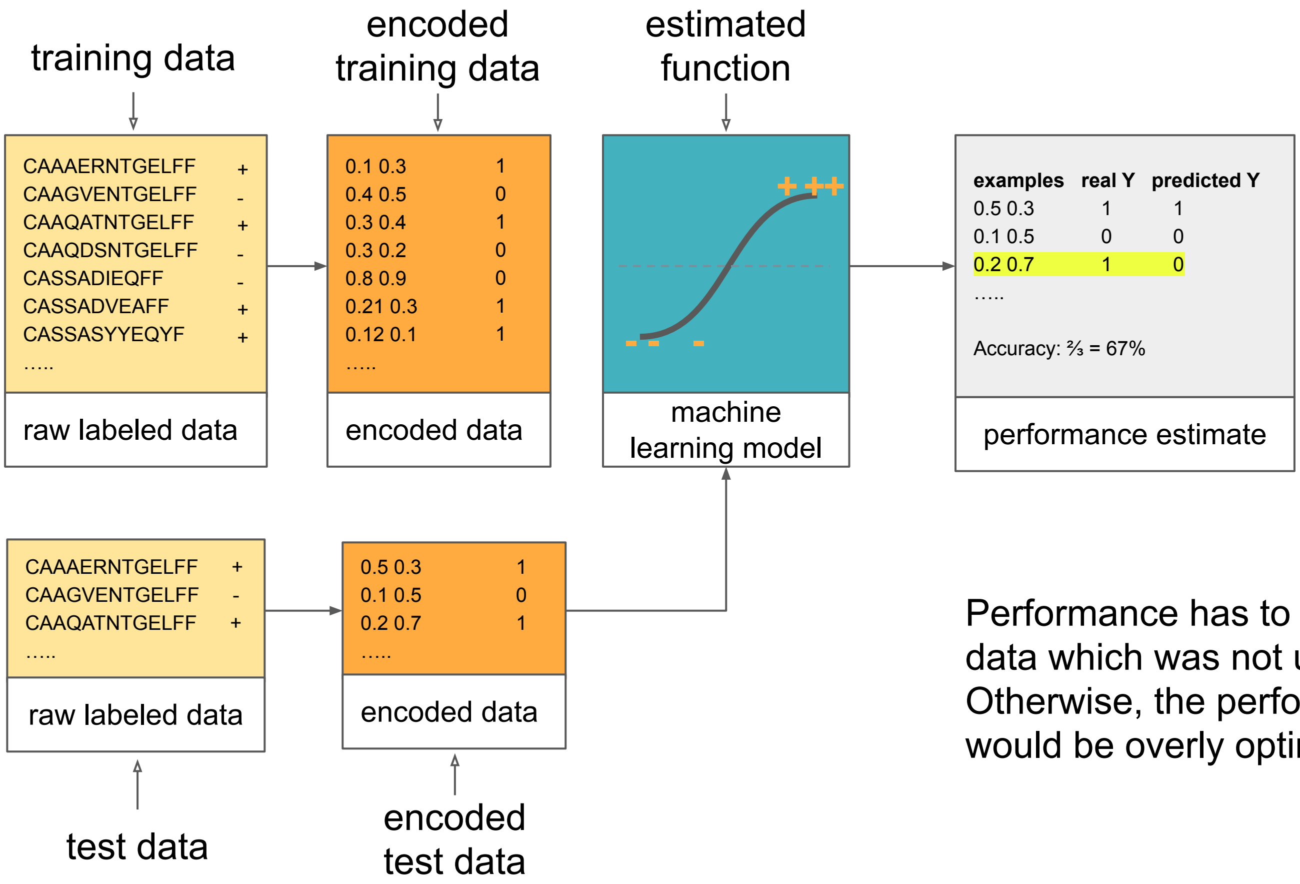
# Estimating the function (training procedure)



# Estimating the function (training procedure)



# Estimating the function (training procedure)




Performance has to be estimated on data which was not used during training. Otherwise, the performance estimate would be overly optimistic.

# Performance metrics - regression

- Depends on the problem and the data
- Regression (label values are continuous values):

Mean squared error (MSE) is among most common metrics:

$$MSE = \frac{1}{\boxed{m}} \sum_{i=1}^{m} (\boxed{y^{(i)}} - \boxed{\hat{y}^{(i)}})^2$$

number of examples

true value of the label for example *i*

predicted value of the label for example *i*

Other regression metrics: mean absolute error, $R^2$

# Performance metrics - classification

- Depends on the problem and the data
- Classification (label values come from a discrete set):

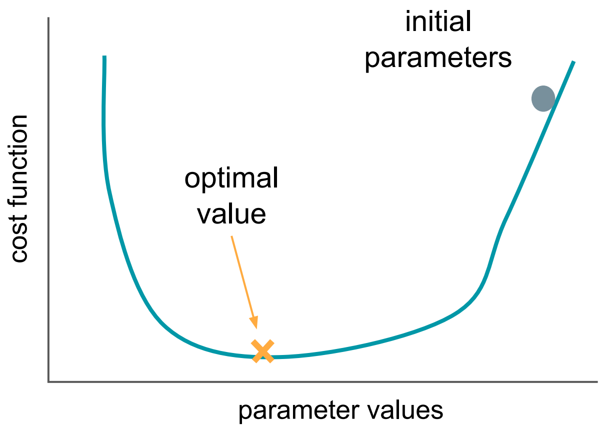
for binary classification, this equality holds:

$$accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ predictions} = \frac{TP + TN}{\underbrace{\boxed{TP}}_{\text{true positives}} + \underbrace{\boxed{TN}}_{\text{true negatives}} + \underbrace{\boxed{FP}}_{\text{false positives}} + \underbrace{\boxed{FN}}_{\text{false negatives}}}$$

Other metrics: balanced accuracy, precision, recall, sensitivity, specificity, ROC curve, AUC, cross-entropy

# Training the machine learning model

- We want to minimize the cost function
- For instance, we can use optimization algorithm called gradient descent:

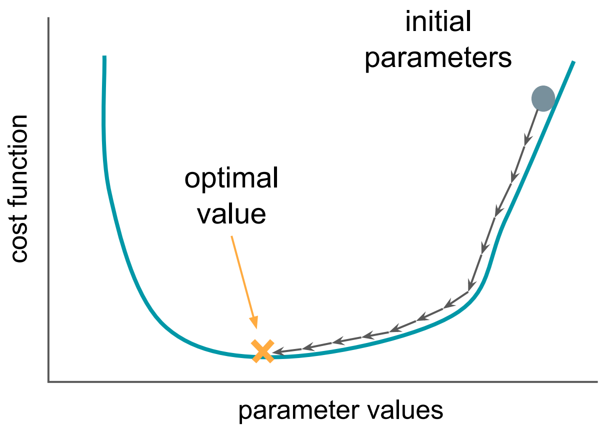



Repeat until optimal solution / max number of iterations:

1. Find derivative of the cost function w.r.t. each of the parameters of the model
2. Update each parameter incrementally using the cost function as a starting point for the update computation

# Training the machine learning model

- We want to minimize the cost function
- For instance, we can use optimization algorithm called gradient descent

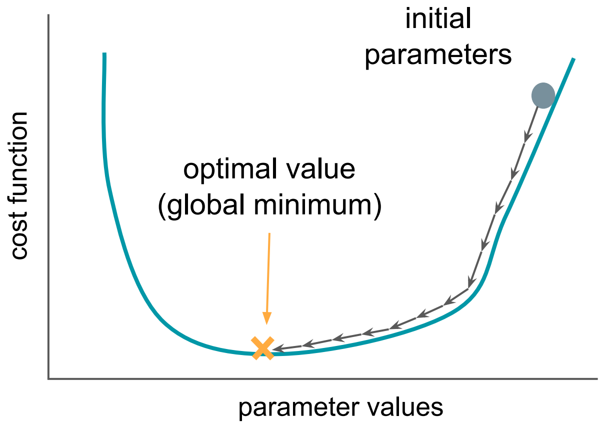


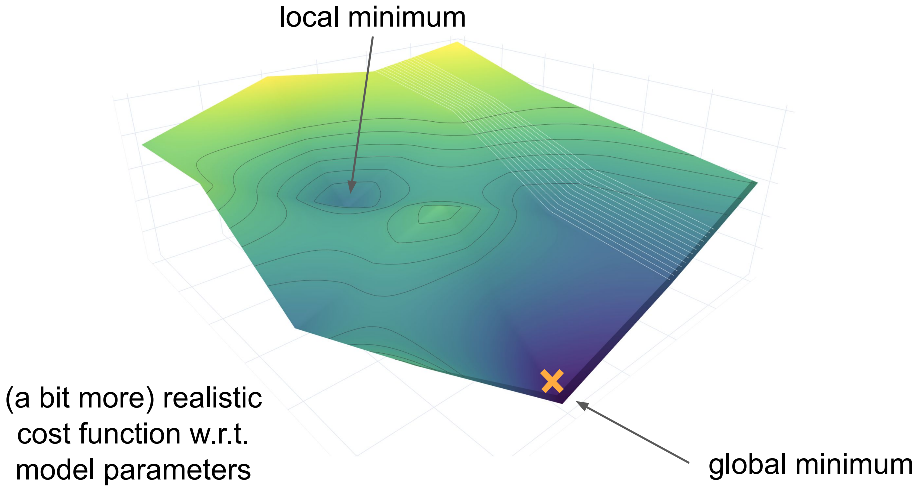

Repeat until optimal solution / max number of iterations:

1. Find derivative of the cost function w.r.t. each of the parameters of the model
2. Update each parameter incrementally using the cost function as a starting point for the update computation

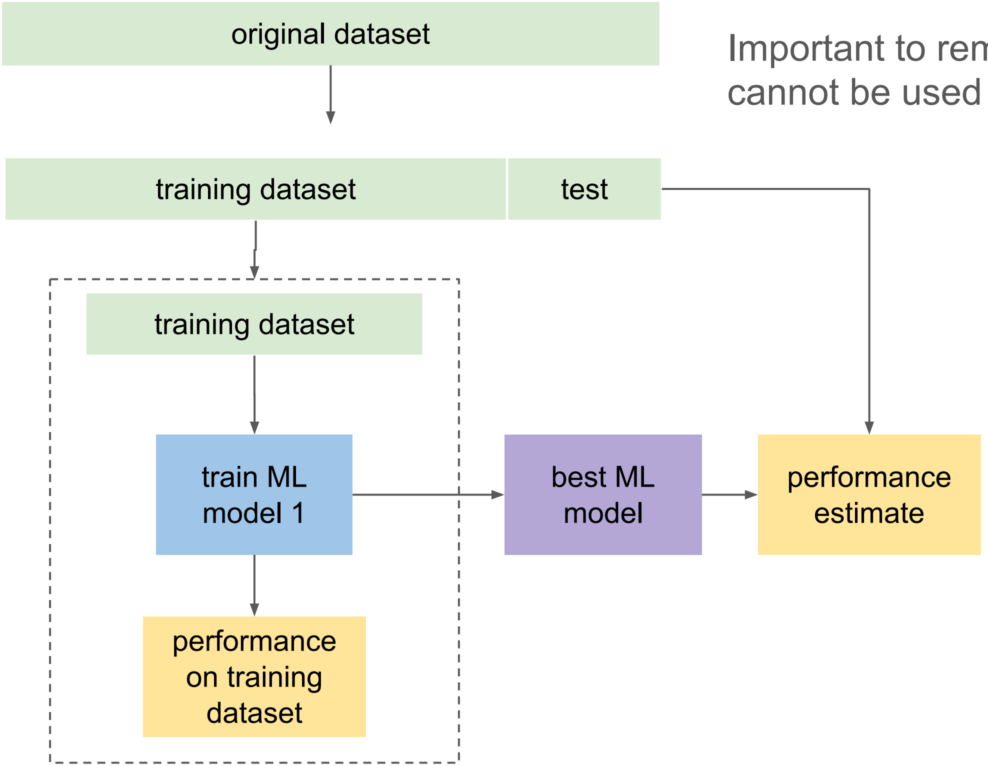
# Training the machine learning model

- ❏ We want to minimize the cost function
- ❏ For instance, we can use optimization algorithm called gradient descent:



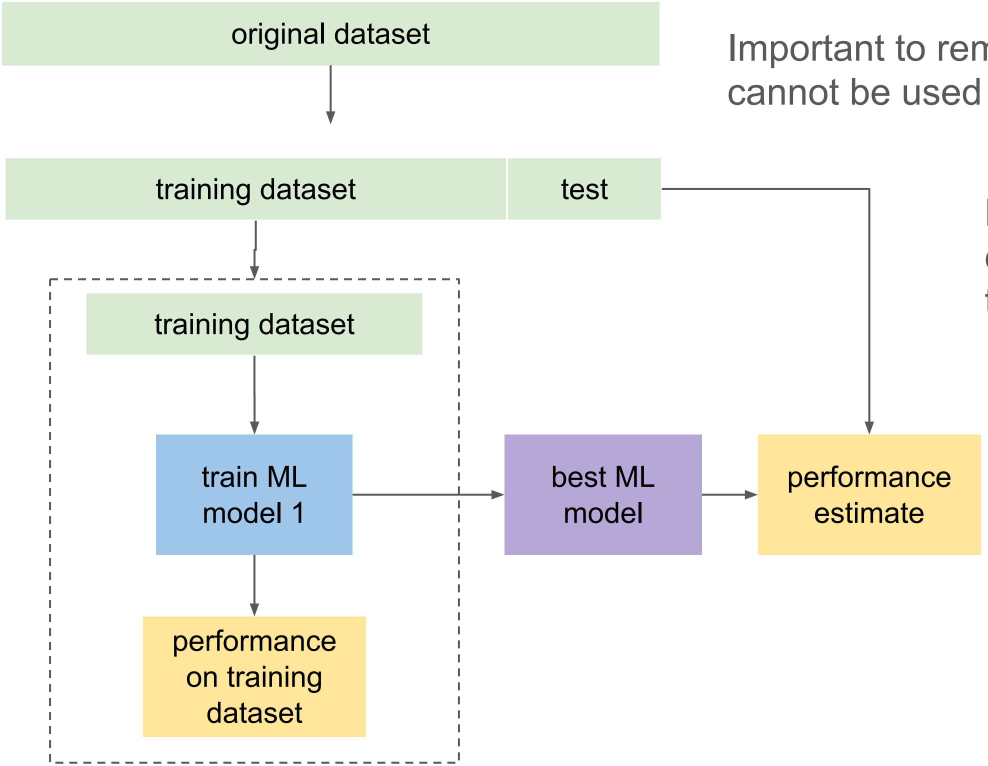
# Machine learning workflow




One way to set up a machine learning workflow

Important to remember: data used to assess the performance cannot be used during training

# Machine learning workflow




One way to set up a machine learning workflow

Important to remember: data used to assess the performance cannot be used during training

Performance on the test data (not seen during training) will typically be worse than performance on validation

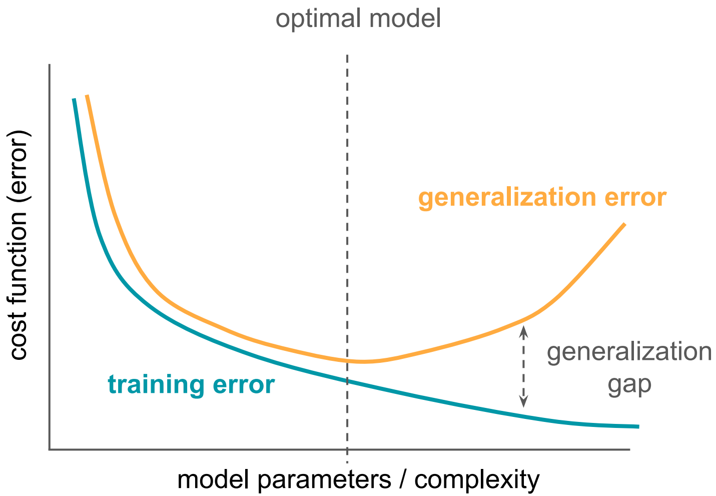
And we will come back to this later...

# Generalization in ML

- ❑ Generalization is the ability of an ML model to perform well on previously unseen data
- ❑ We use error on the test set as an estimate of generalization error
- ❑ Generalization error is the expected error on new data

We want a model which will have:

- ❑ Small error on the training set
- ❑ Small gap between training set error and test set error



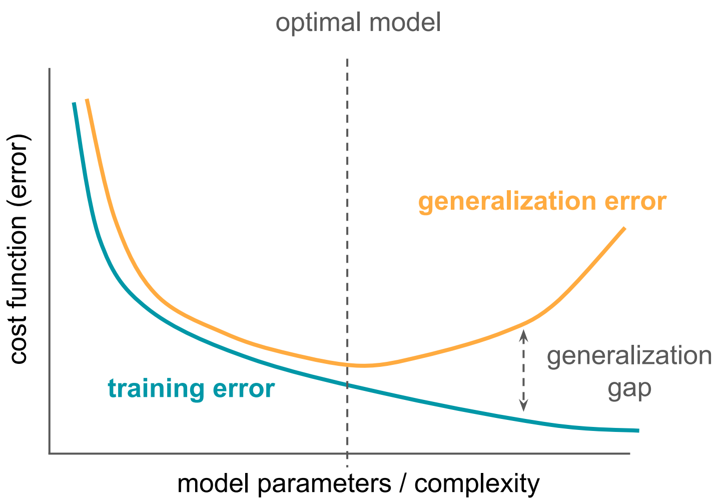
# Generalization in ML

Remember that we can talk about generalization like this only if the i.i.d. assumption at least approximately hold.

- Generalization is the ability of an ML model to perform well on previously unseen data
- We use error on the test set as an estimate of generalization error
- Generalization error is the expected error on new data

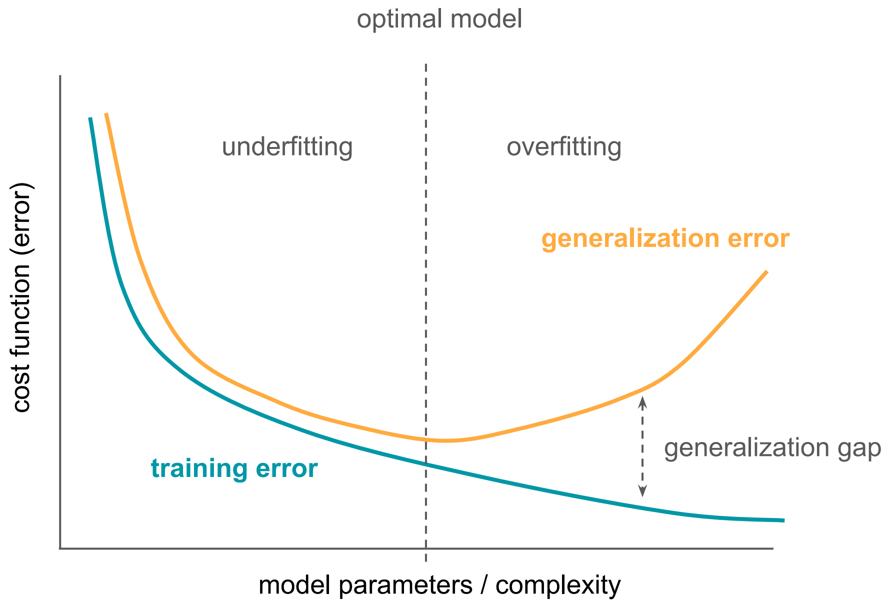
We want a model which will have:

- Small error on the training set
- Small gap between training set error and test set error



# Overfitting and underfitting

- **Underfitting**: the model was not able to learn from the training data - it had high training error
- **Overfitting**: the generalization gap is too large because the model fit the training data too well but failed to extract patterns which would enable good performance on the new (test) data



# References


Goodfellow IJ, Bengio Y, Courville A. *Deep Learning*. MIT Press; 2016. https://mitpress.mit.edu/books/deep-learning

Mitchell T. *Machine Learning*. McGraw Hill; 1997. http://www.cs.cmu.edu/~tom/mlbook.html