

# DEEP LEARNING

## FOR NUCLEAR PHYSICS

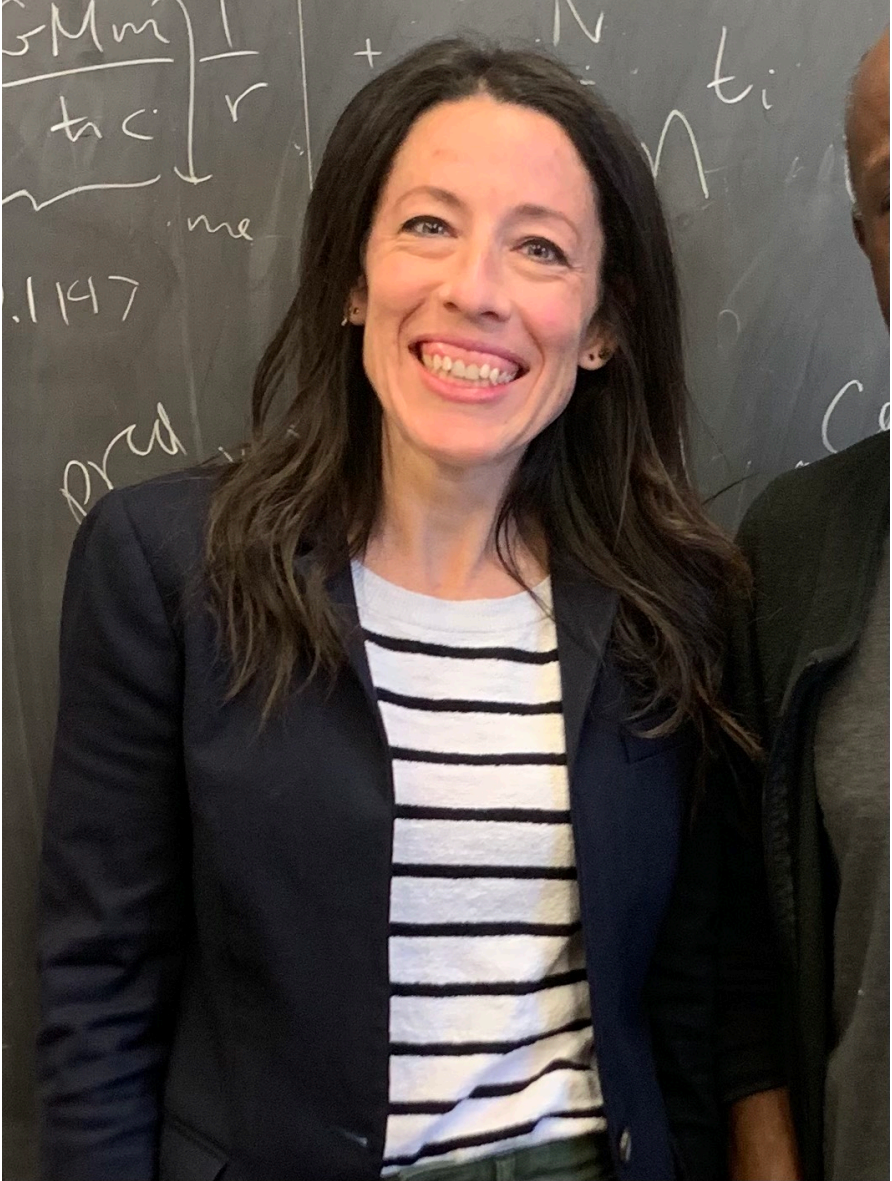
MICHELLE KUCHERA  
DAVIDSON COLLEGE

EXOTIC BEAM SUMMER SCHOOL  
FRIB  
14 JULY 2023

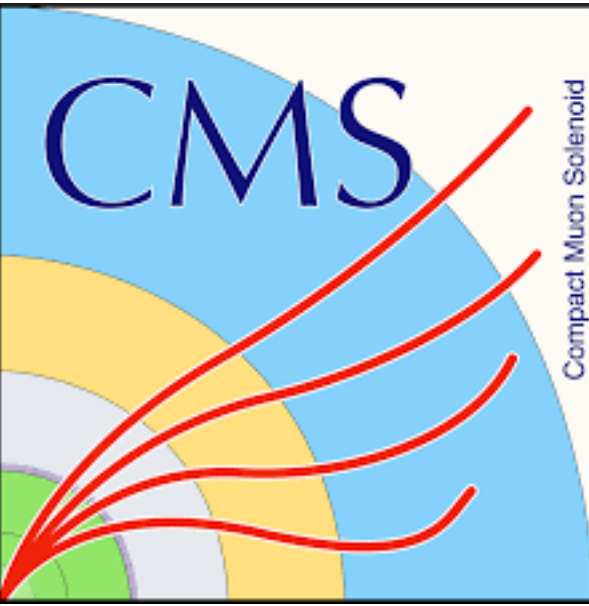
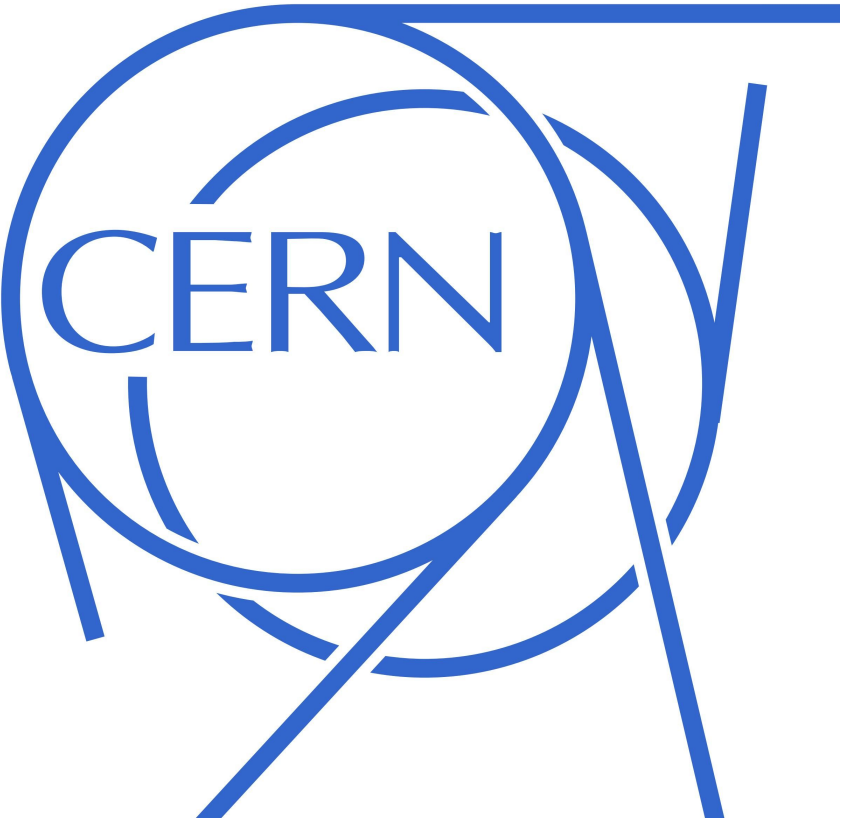
MICHELLE KUCHERA

B.S., M.S. PHYSICS

M.S., PH.D. COMPUTATIONAL SCIENCE



PhD: GPUs for Bayesian Neural Networks (🙄)



# LECTURE TOPICS

- Computational graphs
- Gradient-descent optimization
- Logistic regression
- Deep neural networks
- Learning tasks
- Best practices

**MICHELLE KUCHERA**  
**DAVIDSON COLLEGE**

**EXOTIC BEAM SUMMER SCHOOL**  
**FRIB**  
**14 JULY 2023**

# GOALS

- Each of us learns something today
- Stop me with any questions

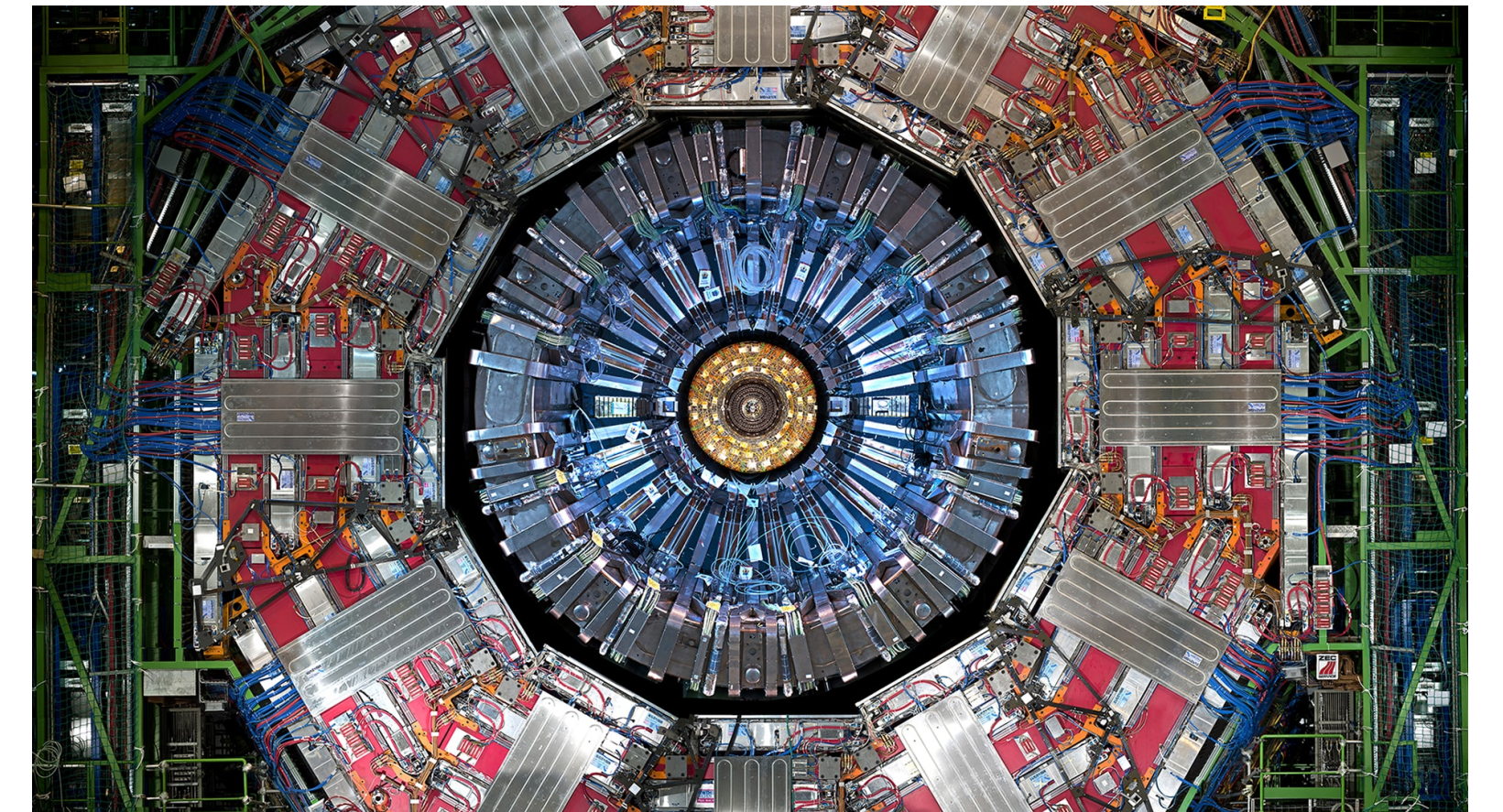
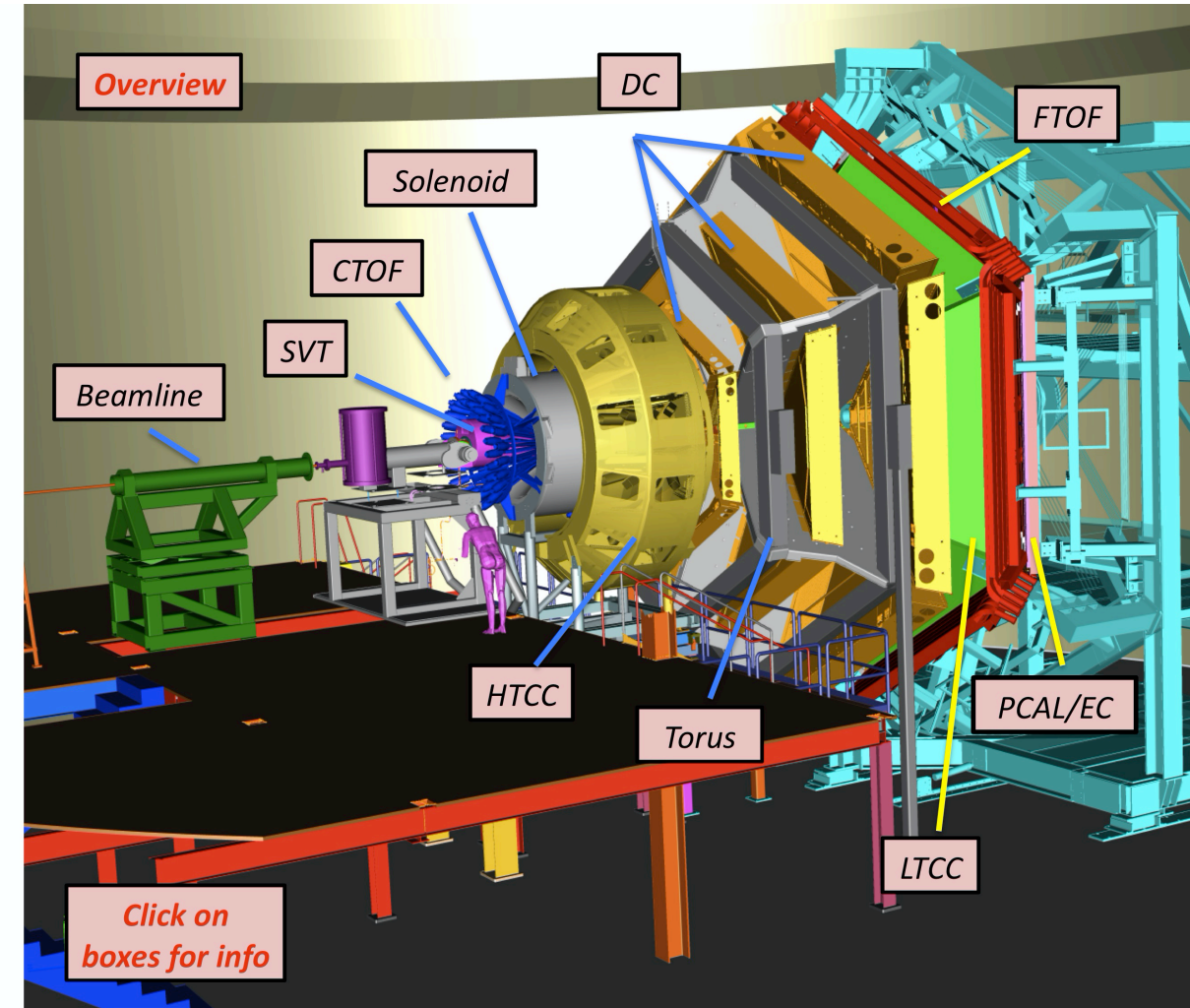
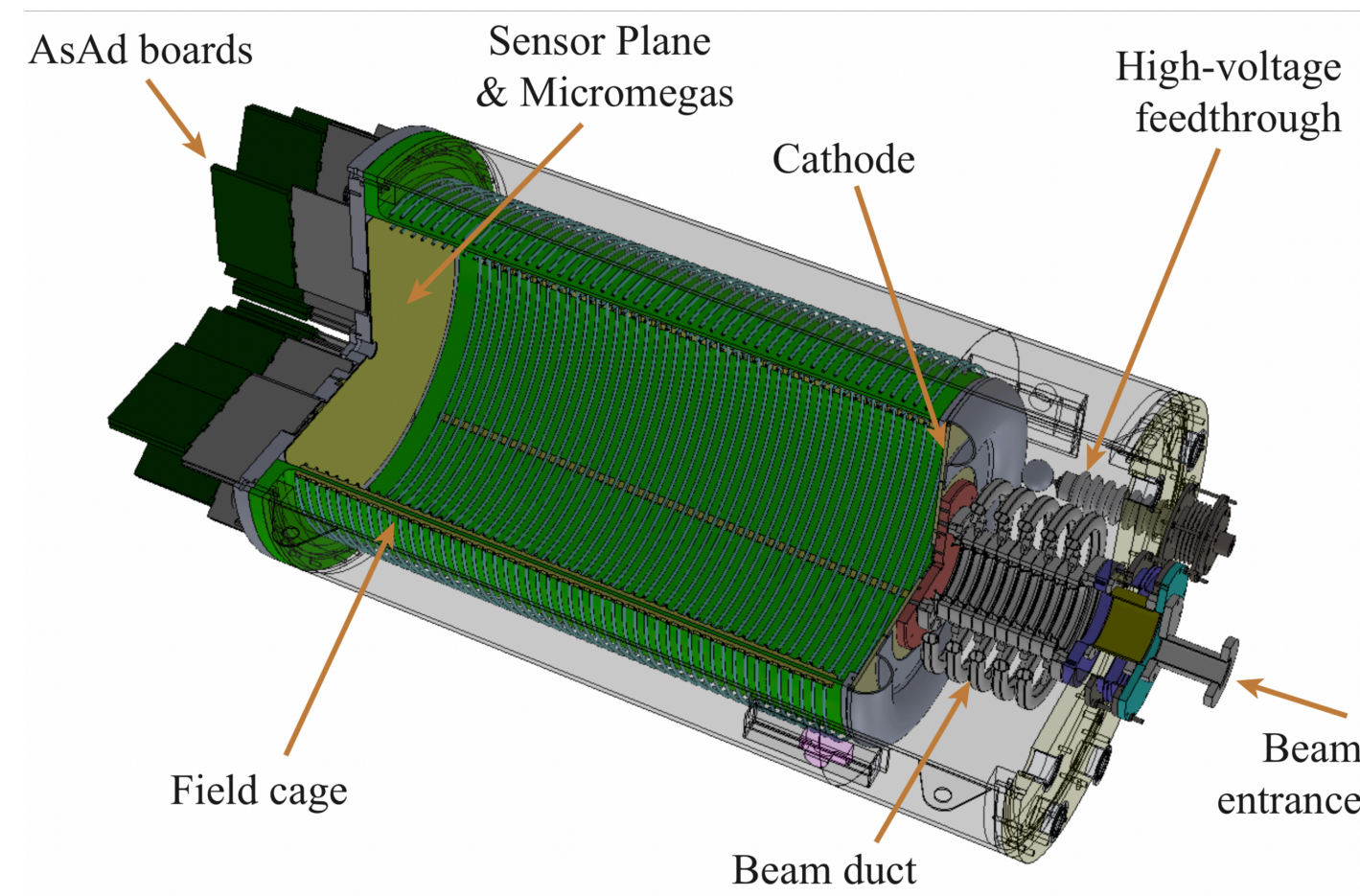
MICHELLE KUCHERA  
DAVIDSON COLLEGE

EXOTIC BEAM SUMMER SCHOOL  
FRIB  
14 JULY 2023

# COMMUNITY

- Each of you arrived here with your own backgrounds, specialty, and path in life
- Your experience and expertise are valuable here, no matter what it is
- If the activity is within your background, help others!
- If you are totally (or a little) lost, ask for help!
- It is our shared goal to have **each** of us leave with some new skill/knowledge/understanding

# EXPERIMENTAL DATA



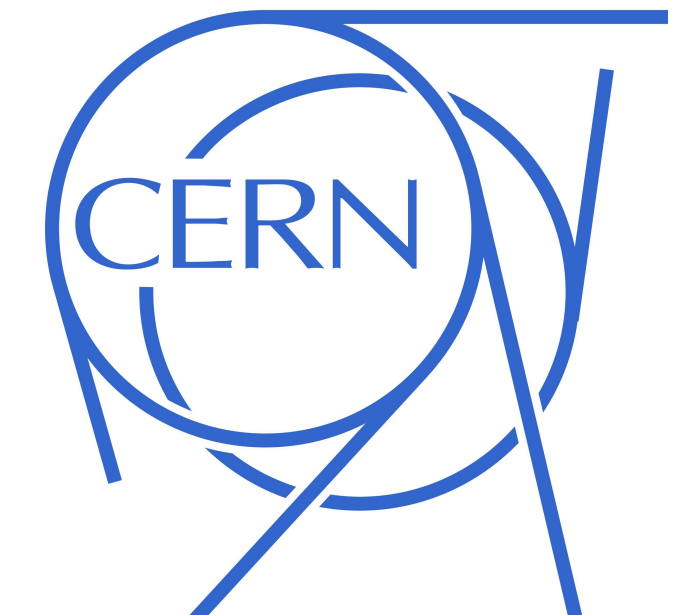
J. BRADT ET. AL., NUCLEAR INSTRUMENTS AND METHODS, 2017.



AT-TPC

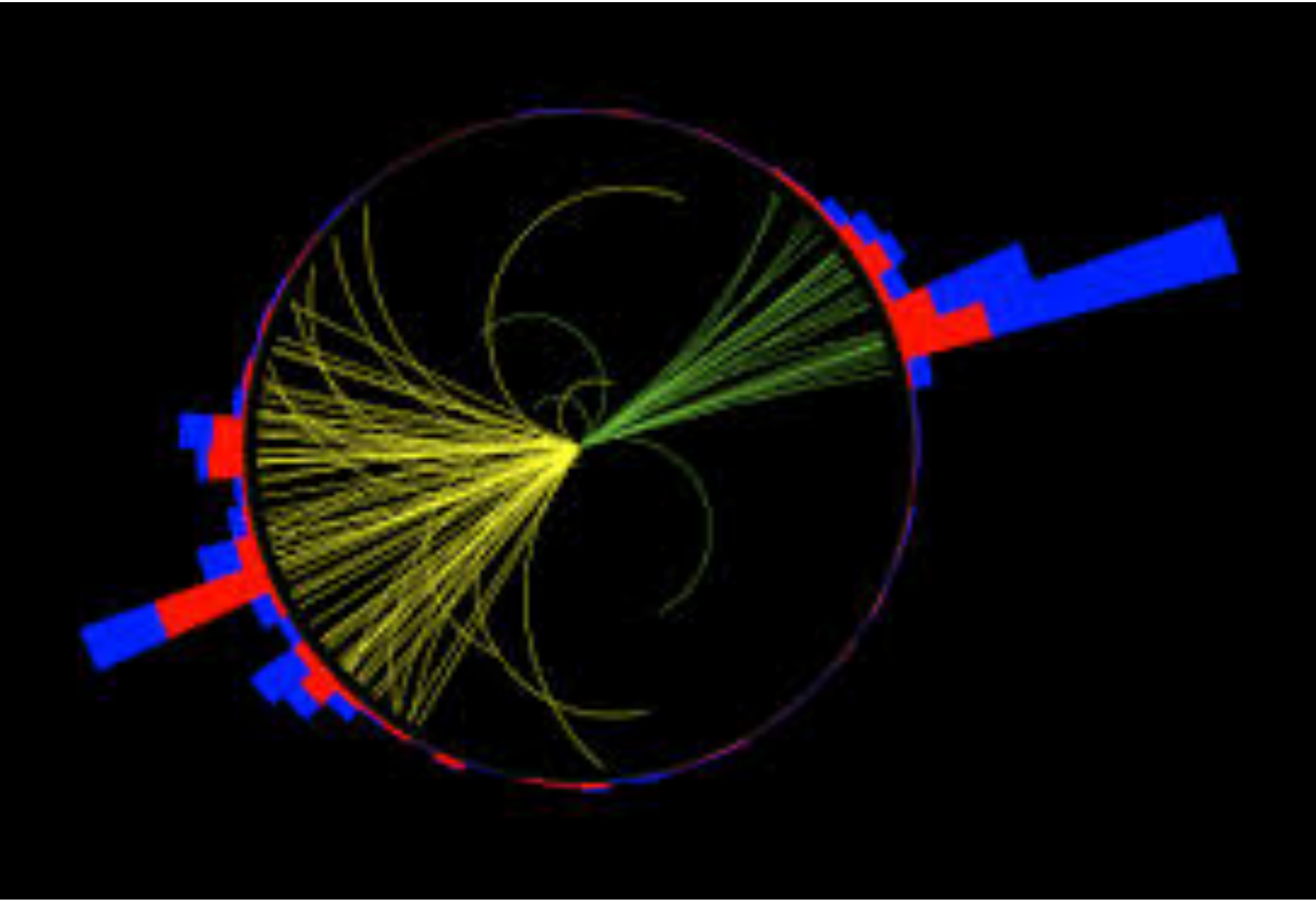
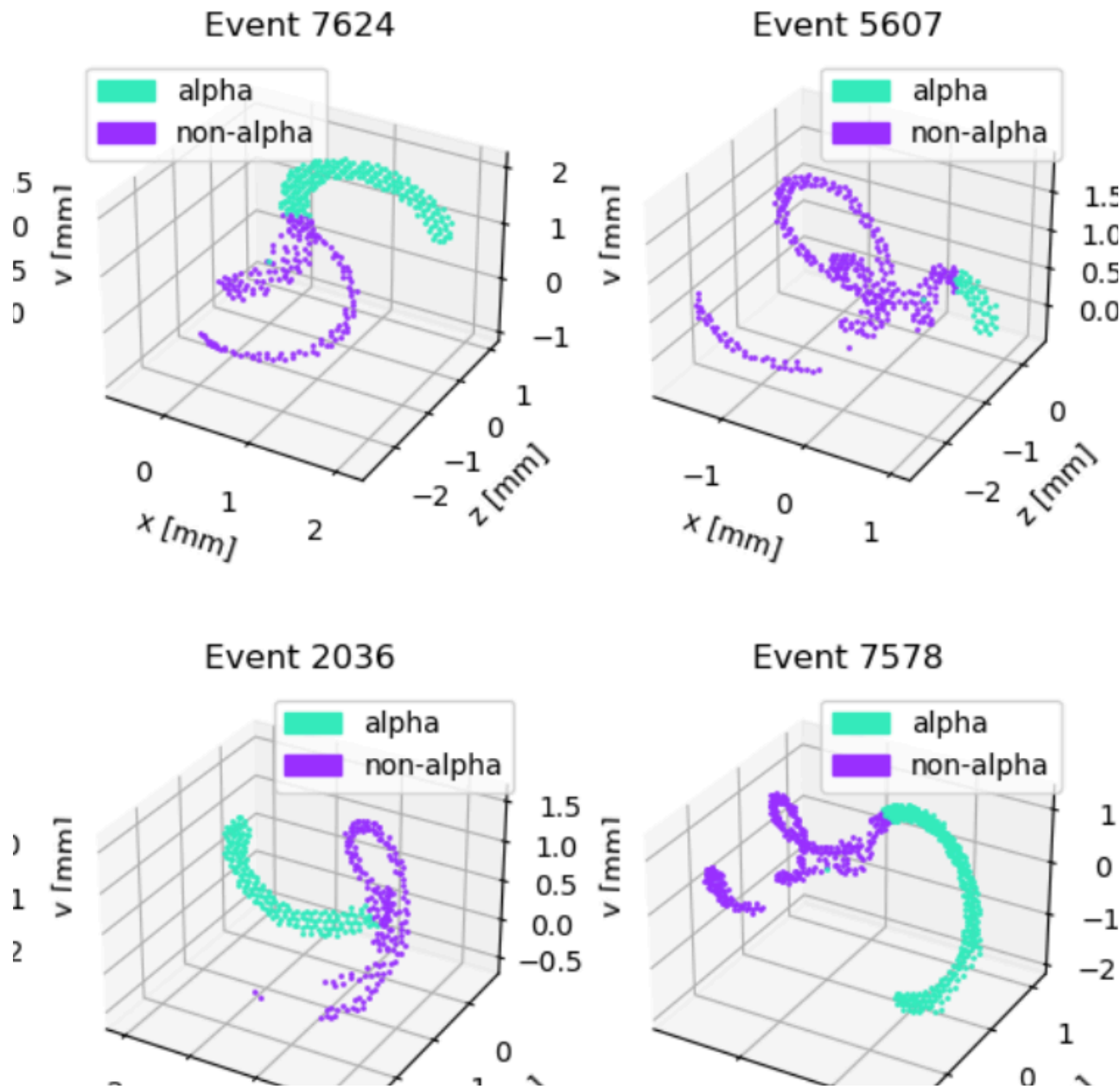


CLAS 12



CMS

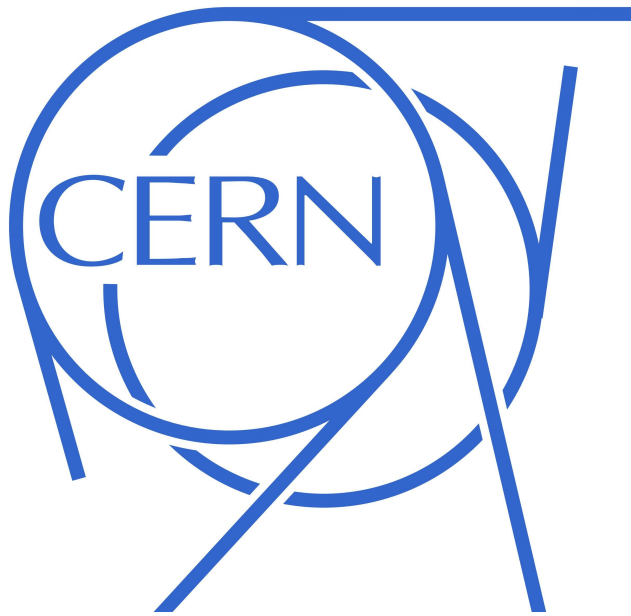
# EXPERIMENTAL DATA



AT-TPC



CLAS 12



CMS

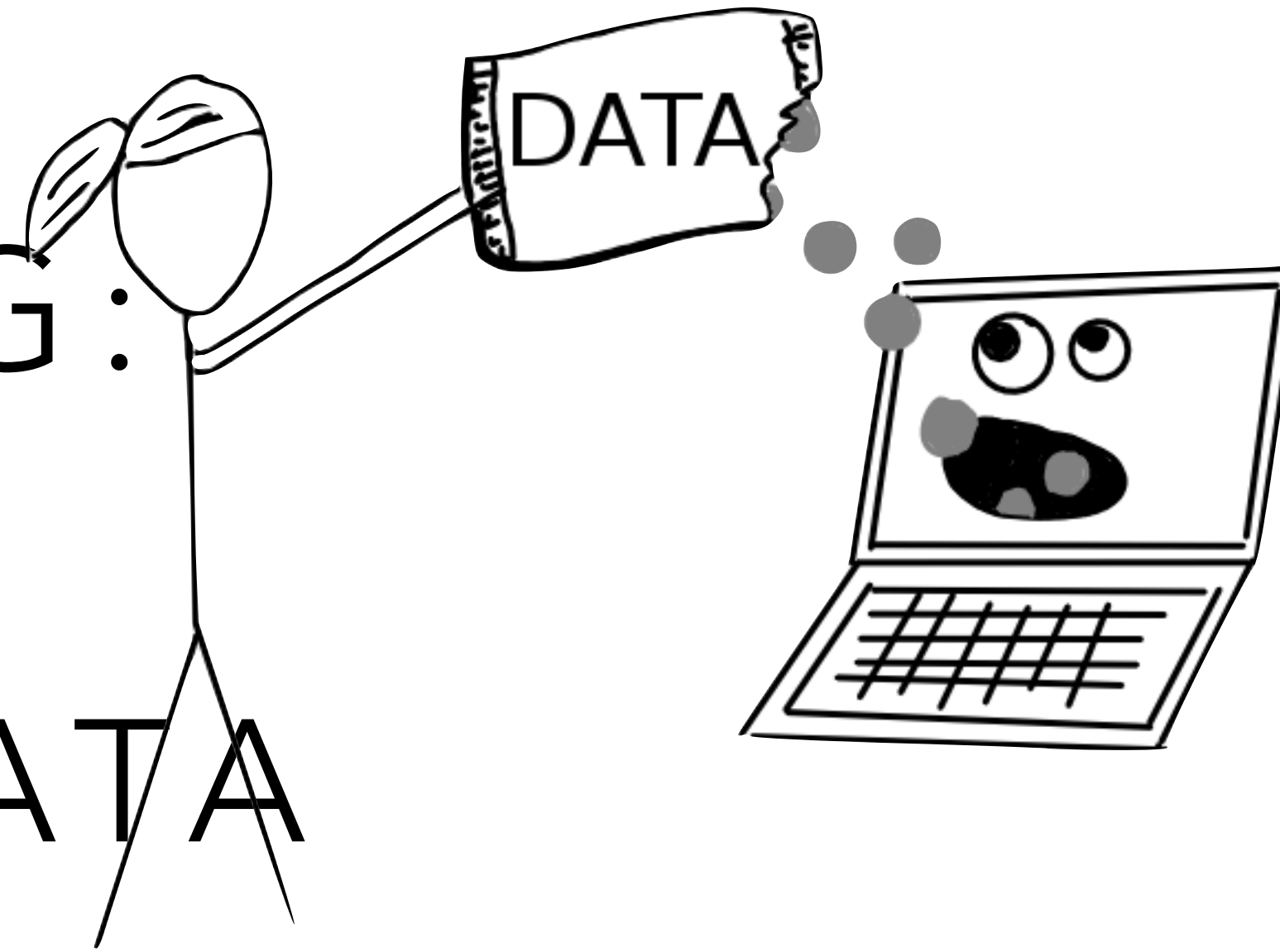
Without Machine Learning

With Machine Learning

MACHINE LEARNING:  
LEARNING FROM DATA



\* VERY SPECIFIC INSTRUCTIONS





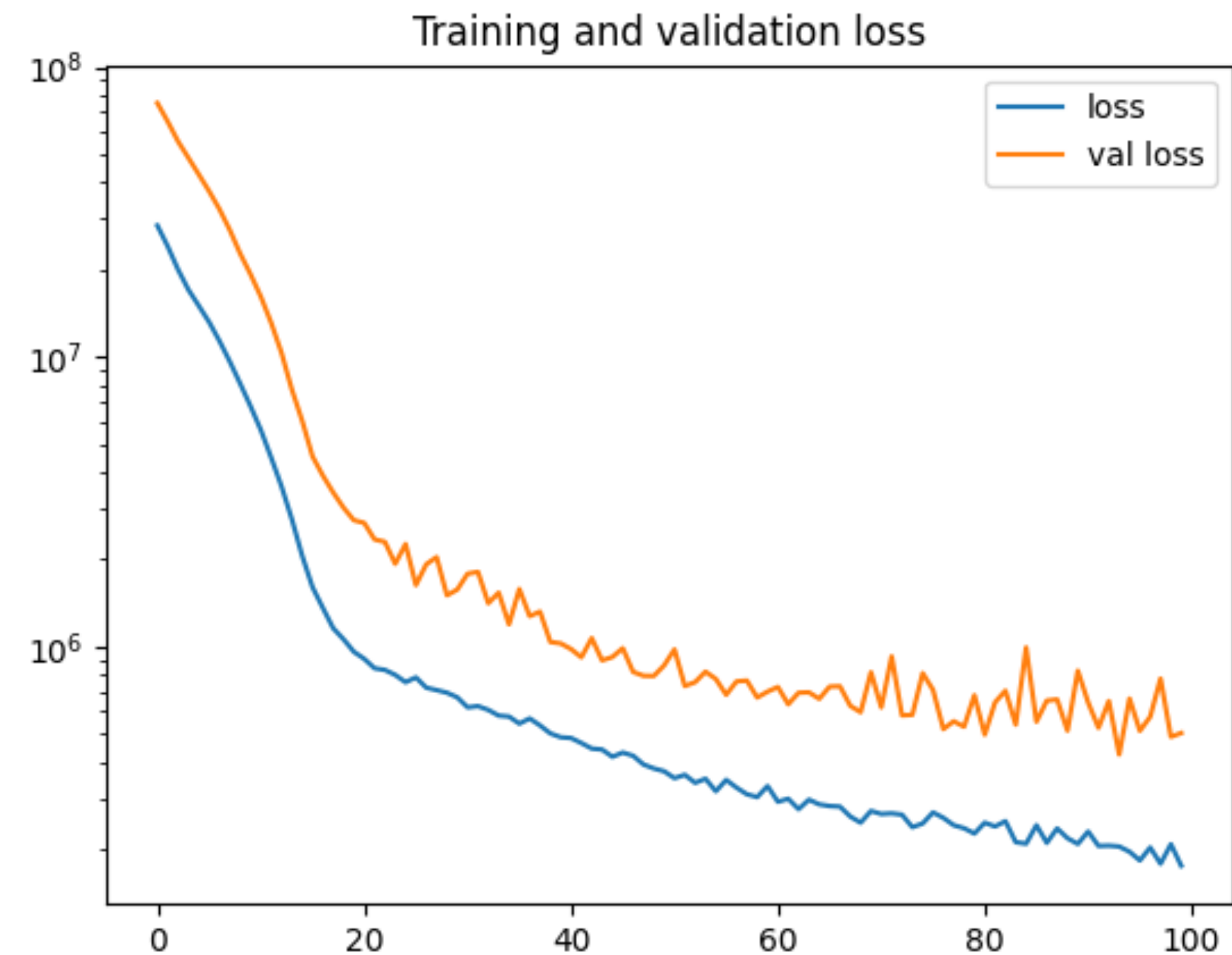
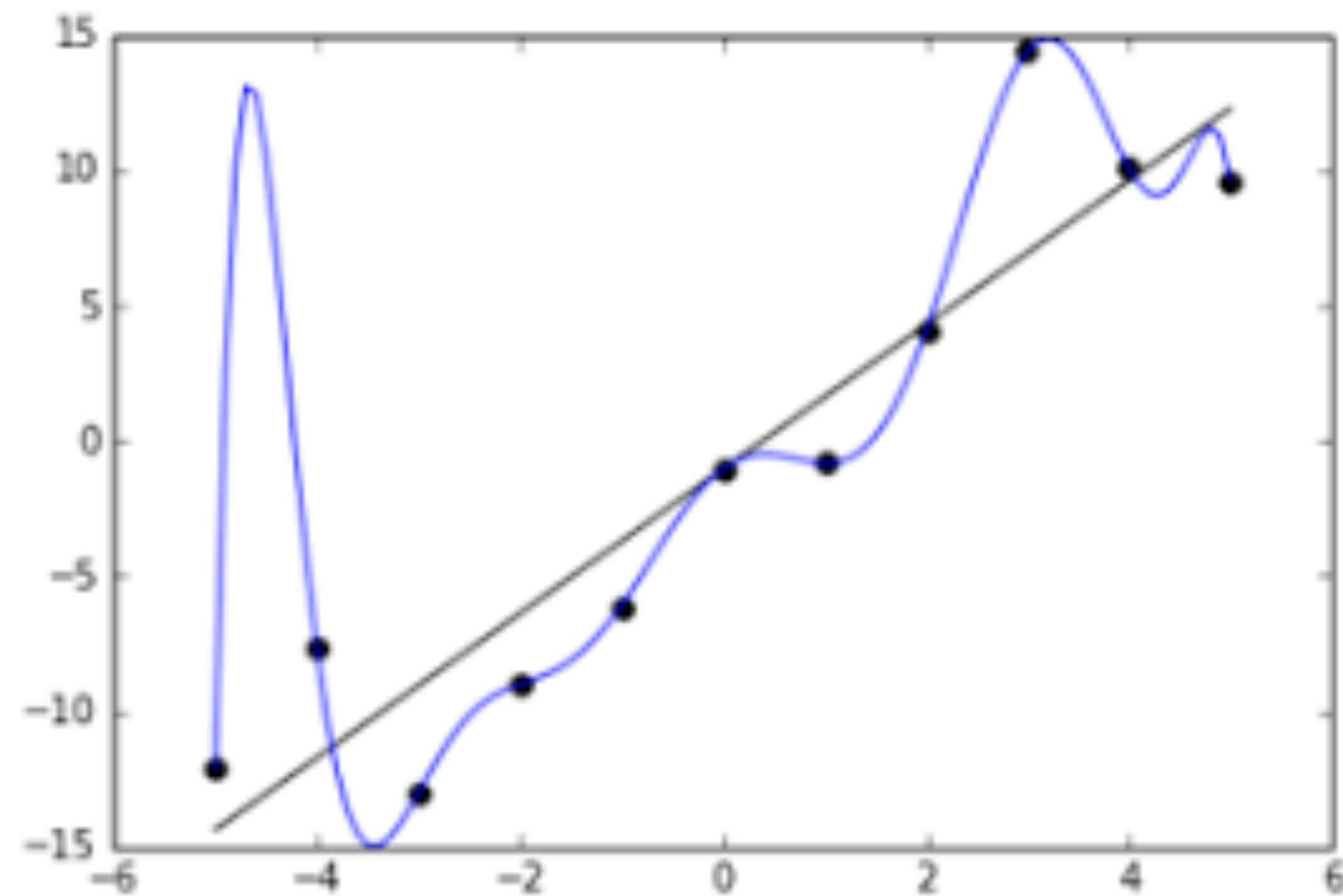
Without Machine Learning

With Machine Learning

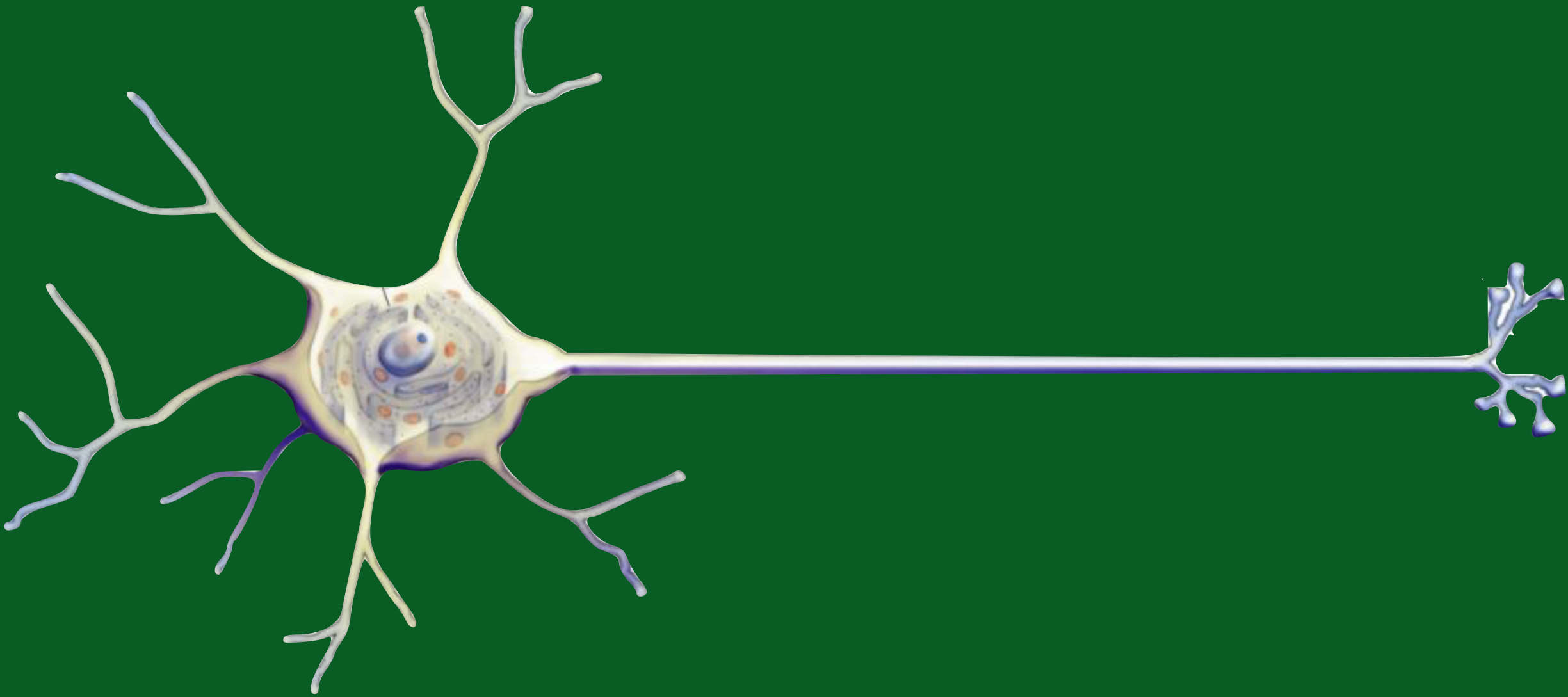
Learning from data is a **paradigm shift** in thinking about predictive models

/ \ \* VERY SPECIFIC INSTRUCTIONS

/ \



# NEURON



# MATHEMATICS





Neural Networks  
Volume 4, Issue 2, 1991, Pages 251-257



## Approximation capabilities of multilayer feedforward networks

Kurt Hornik 

[Show more](#) 

 Share  Cite

[https://doi.org/10.1016/0893-6080\(91\)90009-T](https://doi.org/10.1016/0893-6080(91)90009-T)

[Get rights and content](#)

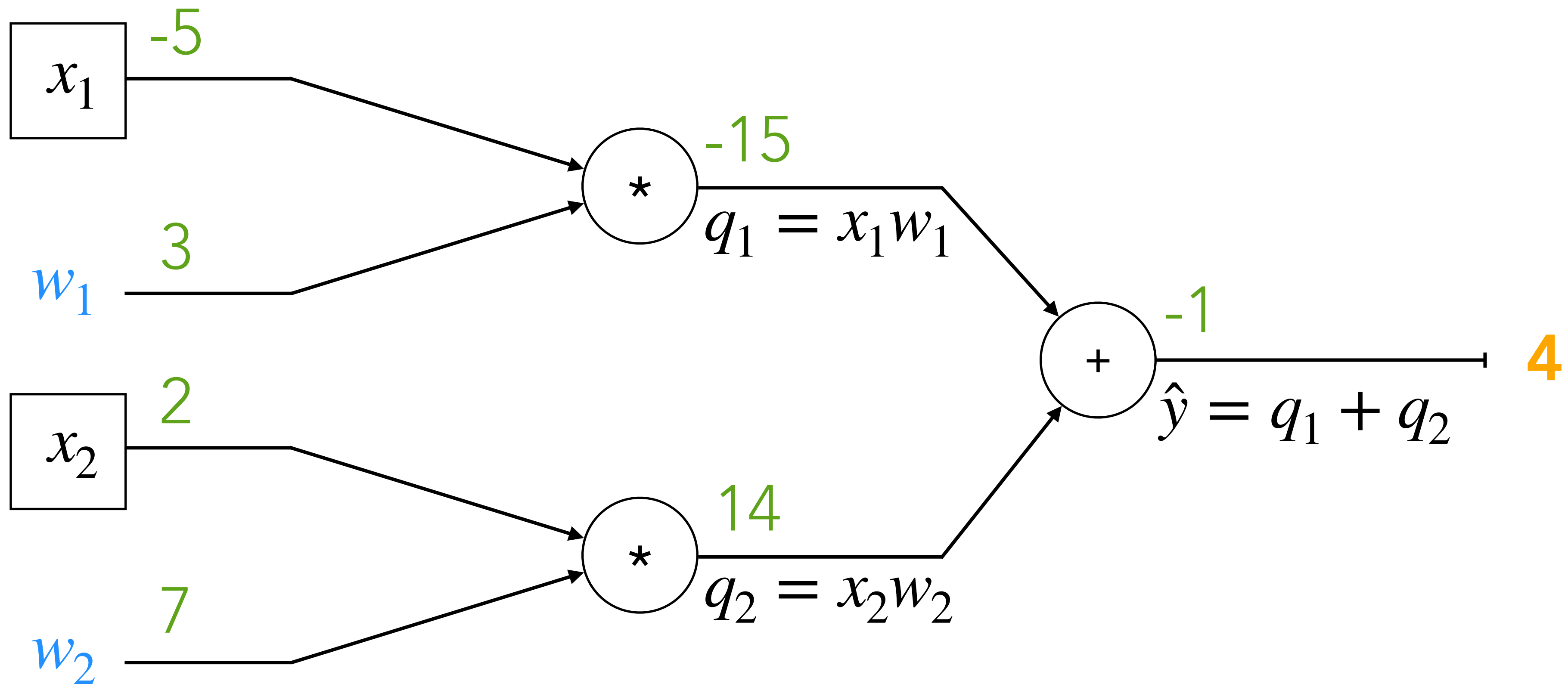
### Abstract

We show that standard multilayer feedforward networks with as few as a single hidden layer and arbitrary bounded and nonconstant activation function are universal approximators with respect to  $L^p(\mu)$  performance criteria, for arbitrary finite input environment measures  $\mu$ , provided only that sufficiently many hidden units are available. If the activation function is continuous, bounded and nonconstant, then continuous mappings can be learned uniformly over compact input sets. We also give very general conditions ensuring that networks with sufficiently smooth activation functions are capable of arbitrarily accurate approximation to a function and its derivatives.

MATHEMATICS

# COMPUTATIONAL GRAPH

$$\hat{y} = x_1 w_1 + x_2 w_2$$

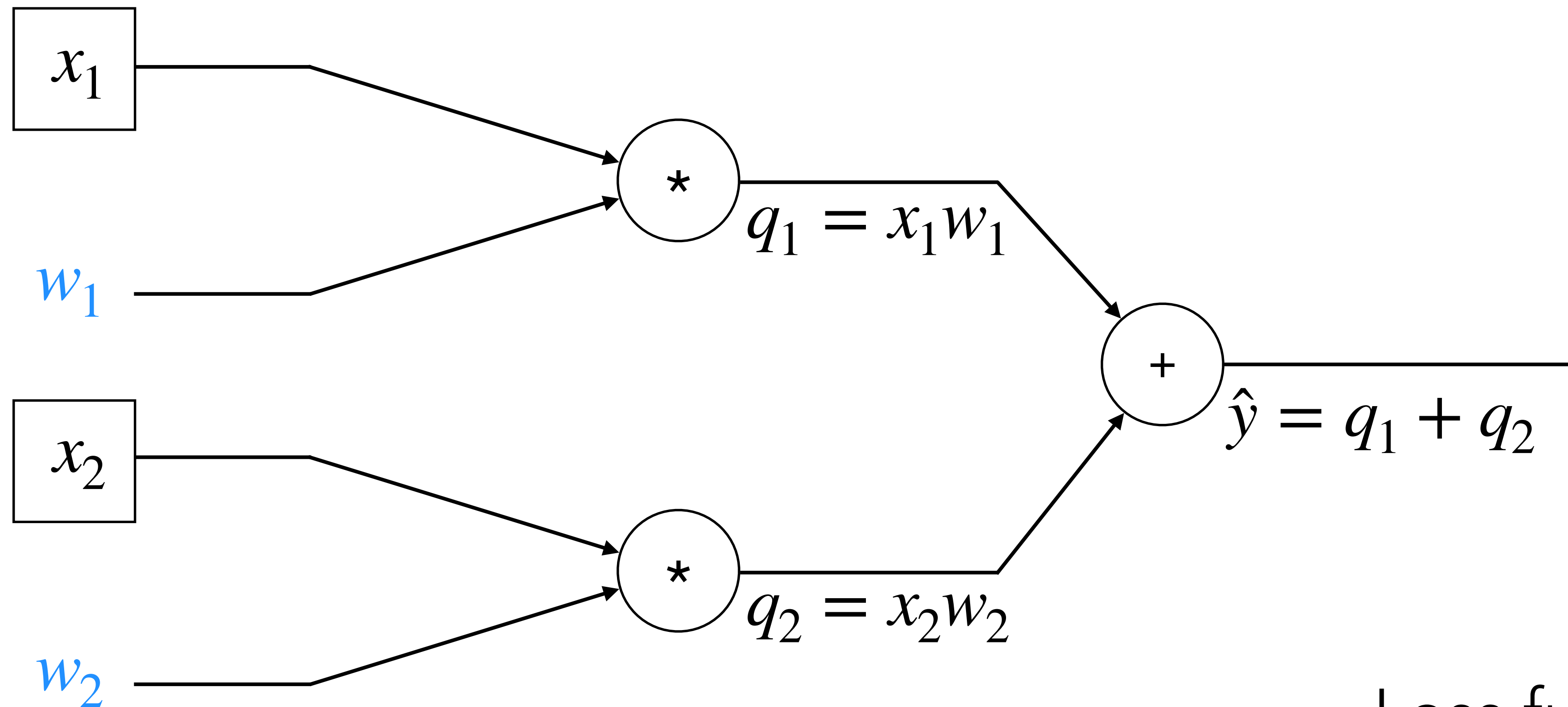


MACHINE LEARNING

SUPERVISED LEARNING

# REGRESSION

$$\hat{y} = x_1 w_1 + x_2 w_2$$

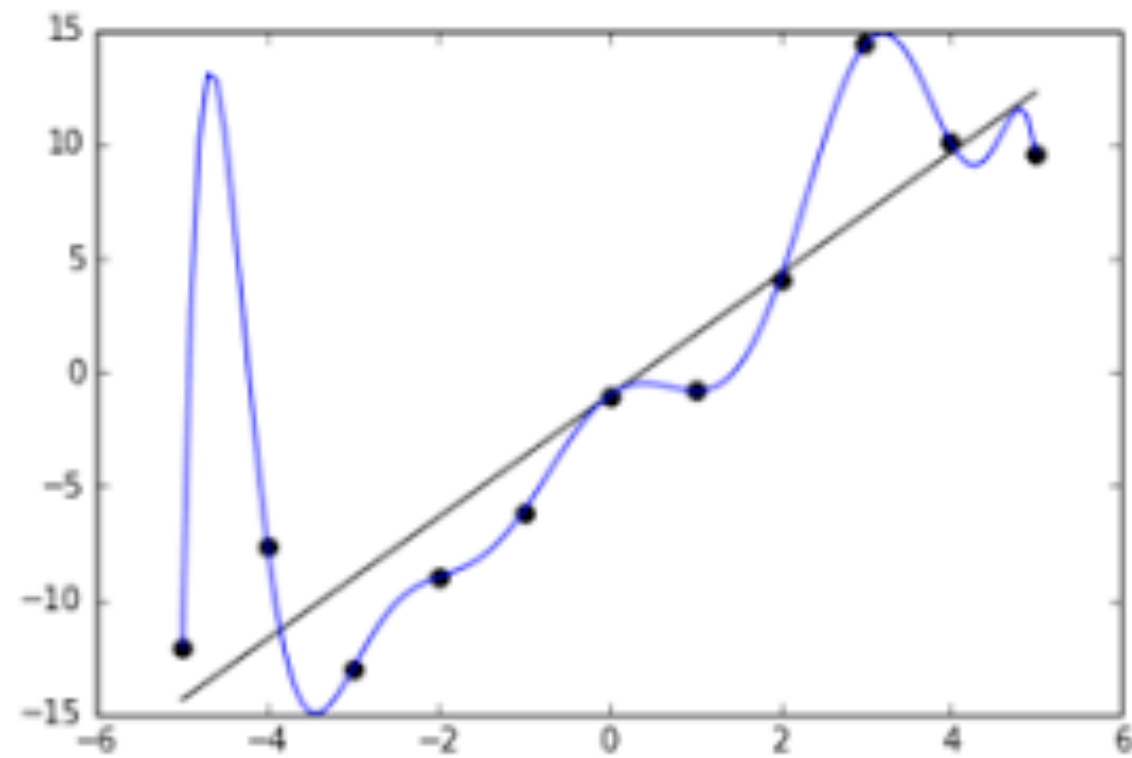
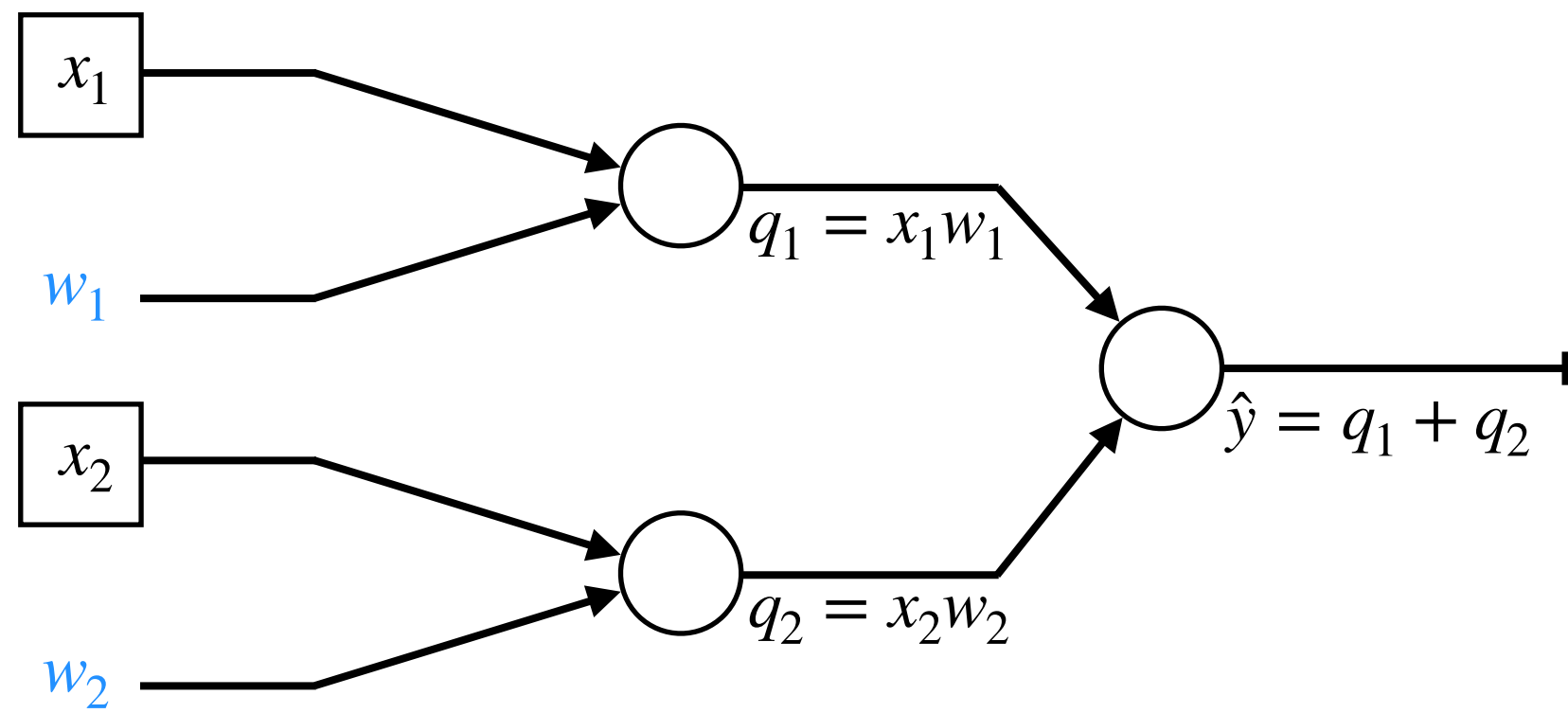


Loss function

$$J(w) = \hat{y} - y$$

# SUPERVISED LEARNING

$$\hat{y} = x_1 w_1 + x_2 w_2$$



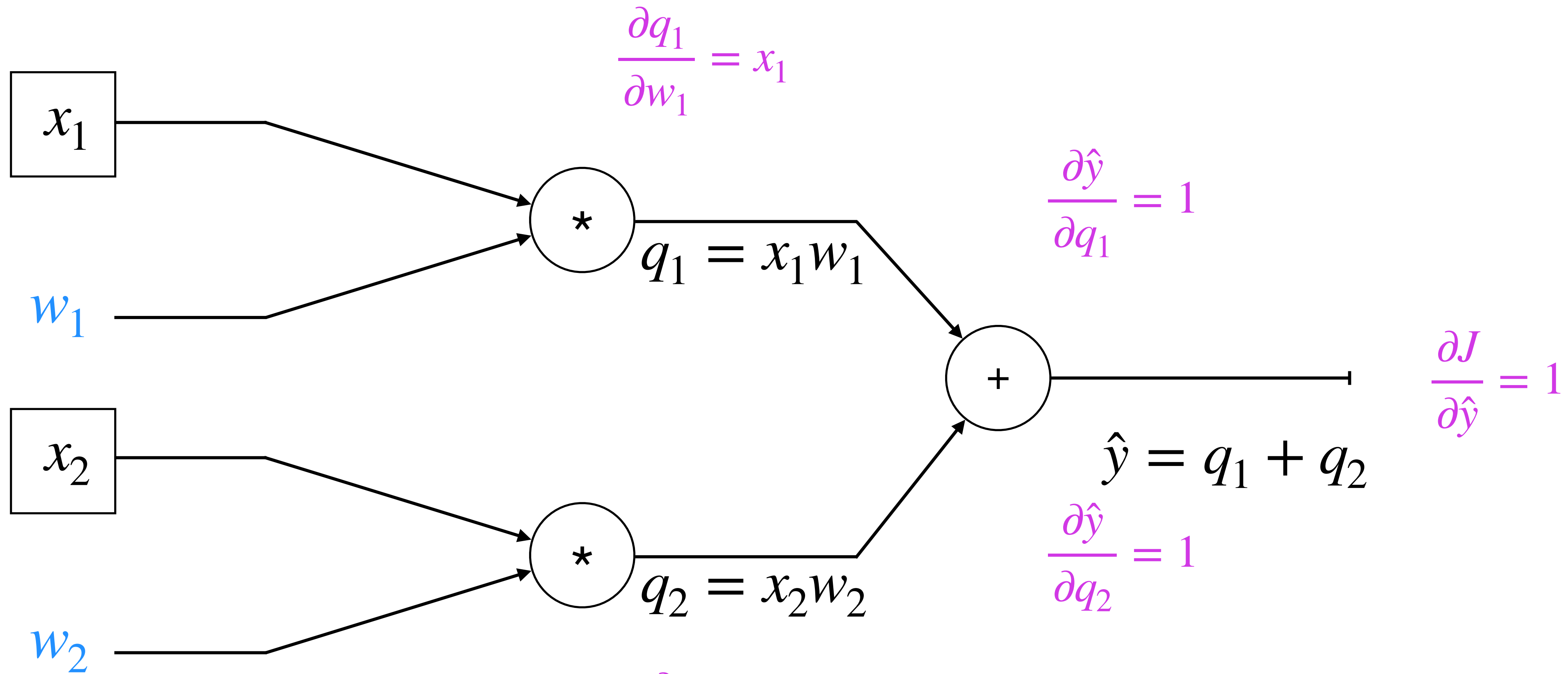
	Feature 1	Feature 2	Feature 3	Target
Example 1				
Example 2				
Example 3				
Example 4				

**Loss function**  
MSE across N examples

$$J(w) = \frac{1}{N} \sum_{i=0}^N (\hat{y}_i - y_i)^2$$

# BACKPROPAGATION

$$w_1 = w_1 - \eta * \frac{\partial J}{\partial \hat{y}} \frac{\partial f}{\partial q_1} \frac{\partial q_1}{\partial w_1}$$



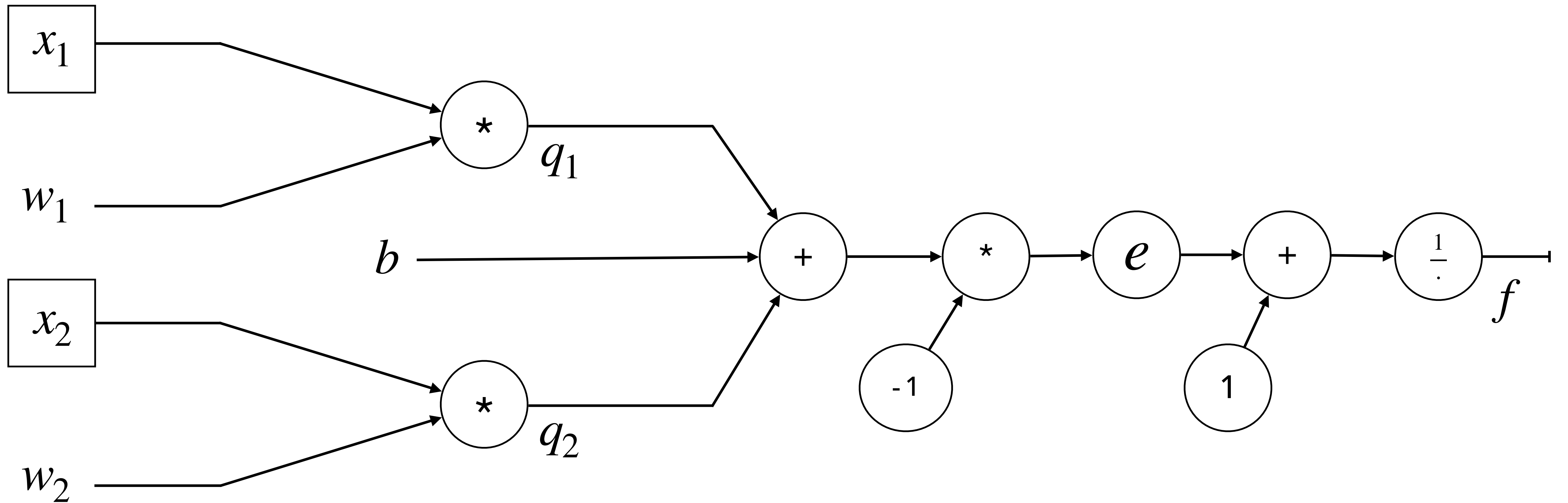
$$w_2 = w_2 - \eta * \frac{\partial J}{\partial \hat{y}} \frac{\partial f}{\partial q_2} \frac{\partial q_2}{\partial w_2}$$

Loss function

$$J(w) = \hat{y} - y$$

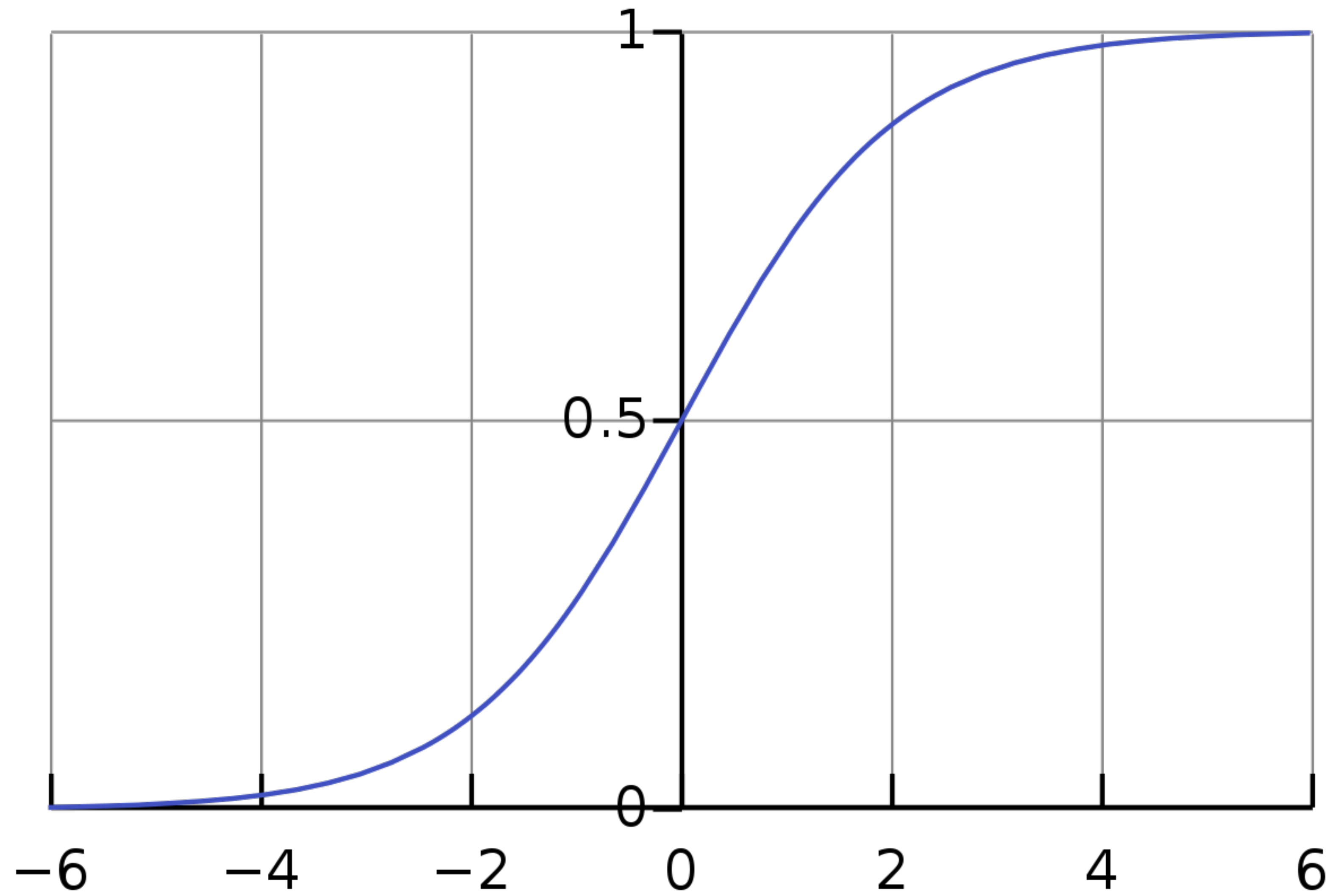


# LOGISTIC REGRESSION

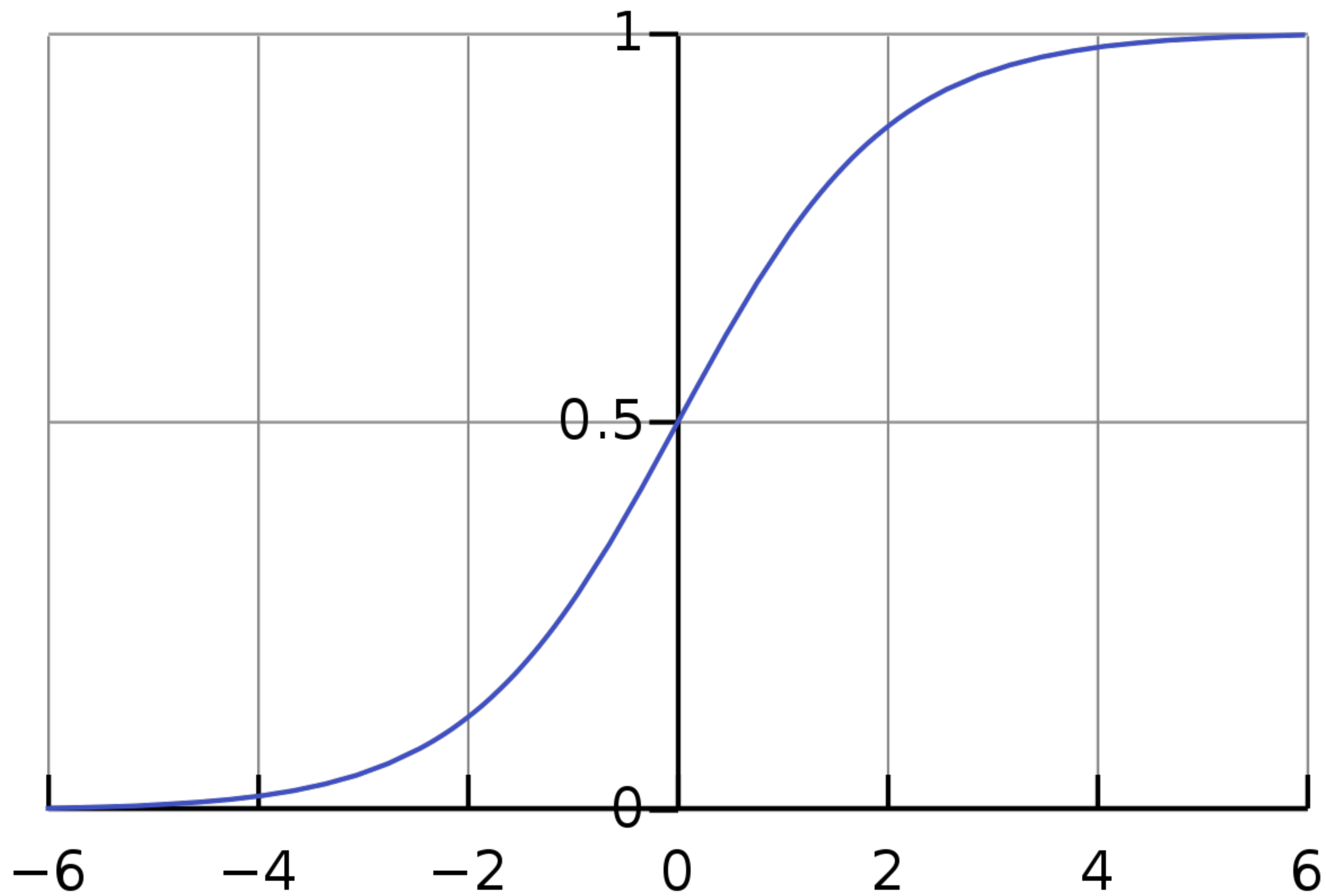


$$f = \frac{1}{1 + e^{-(x_1 w_1 + x_2 w_2 + b)}}$$

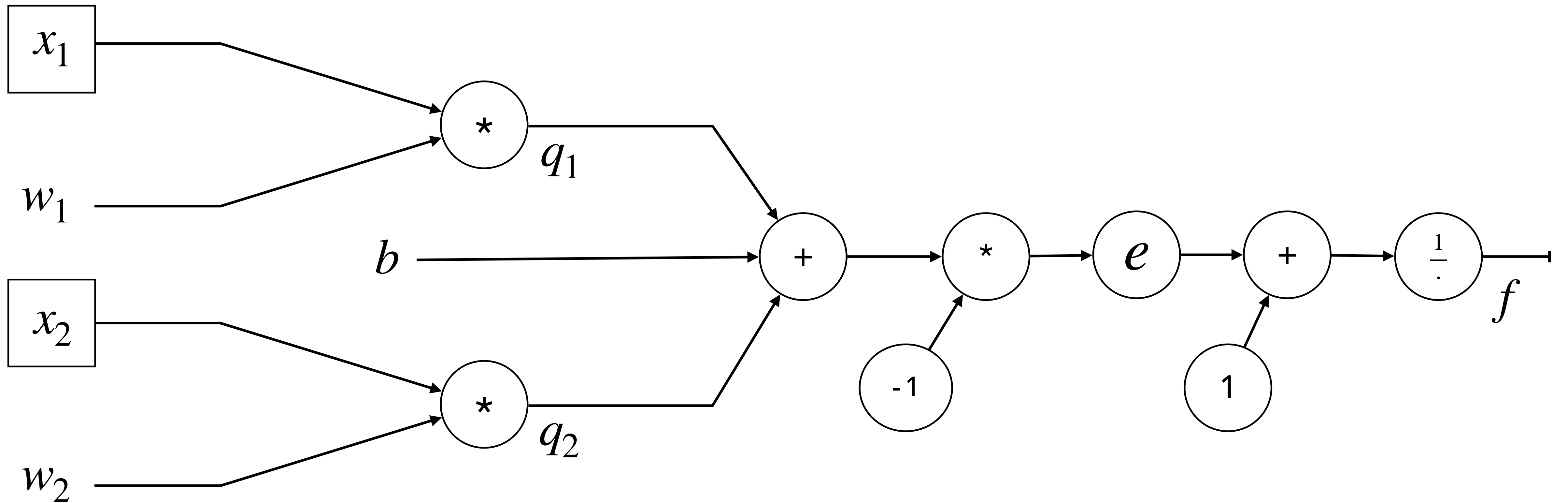
# LOGISTIC REGRESSION



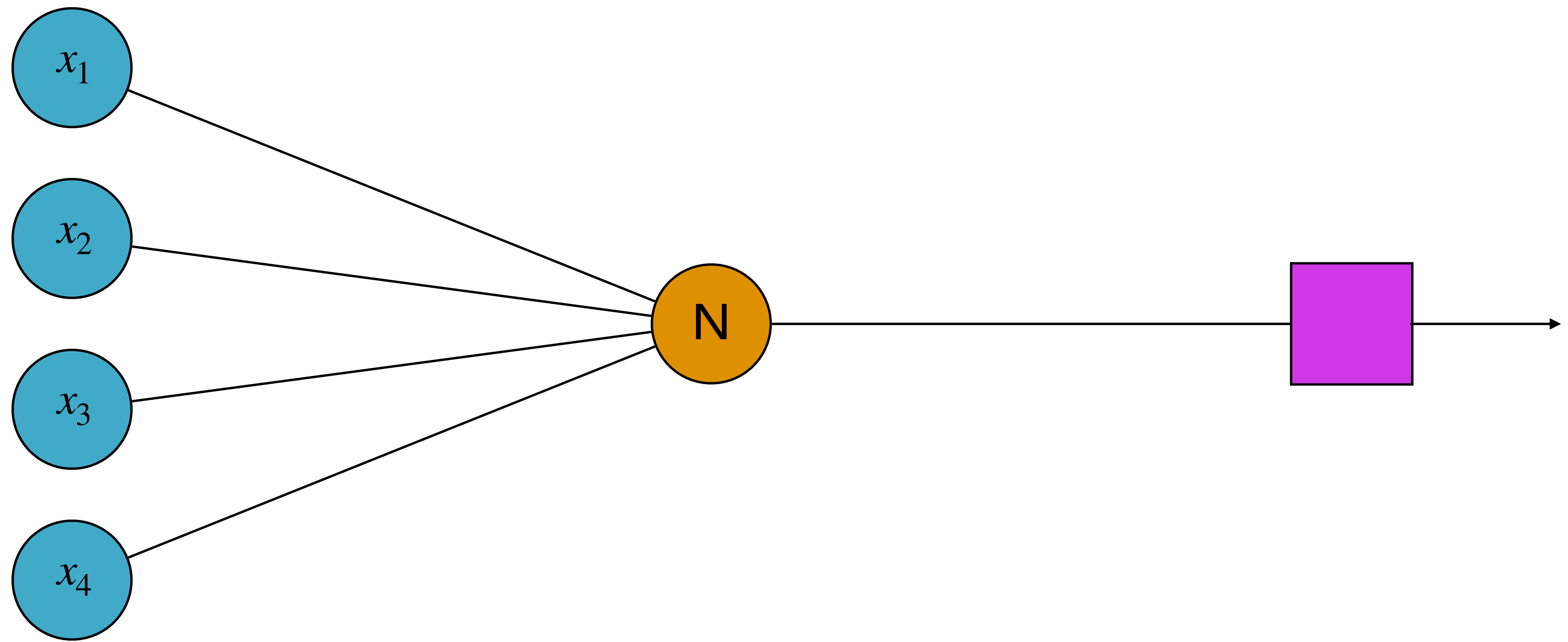
# CLASSIFICATION



# LOGISTIC REGRESSION



$$f = \frac{1}{1 + e^{-(x_1 w_1 + x_2 w_2 + b)}}$$

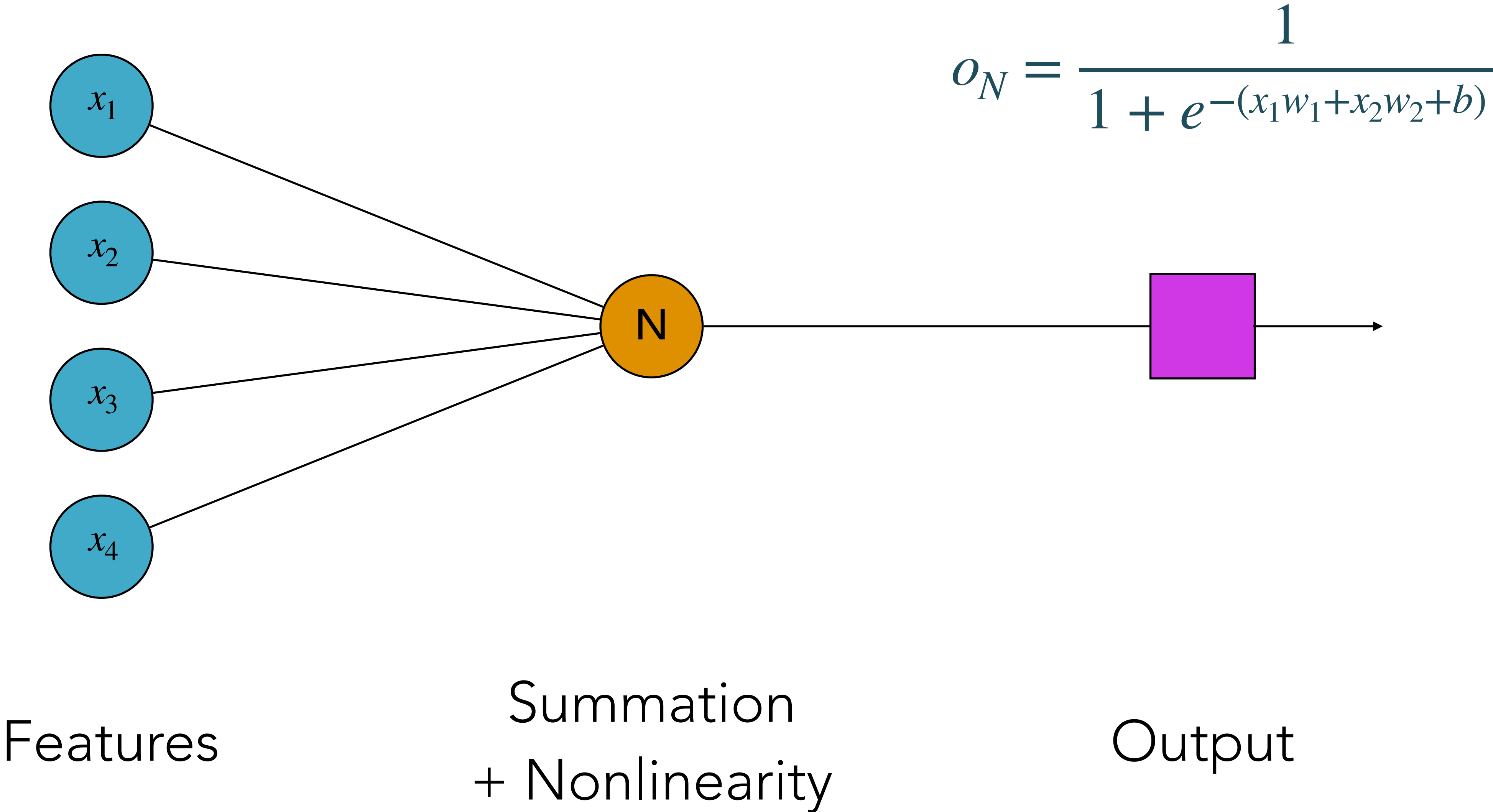


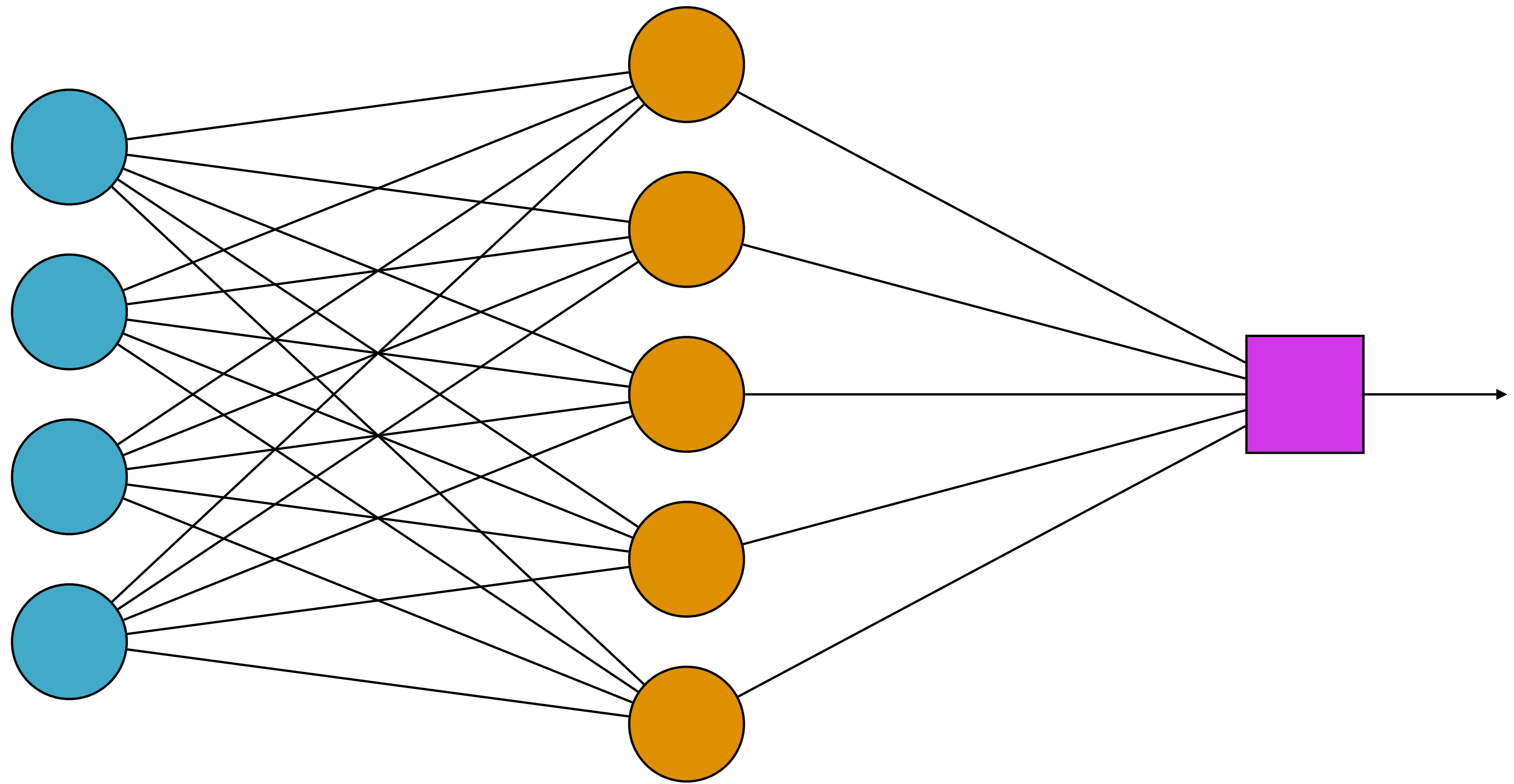
Features

Summation  
+ Nonlinearity

Output

# CHECK: HOW MANY TRAINABLE PARAMETERS?



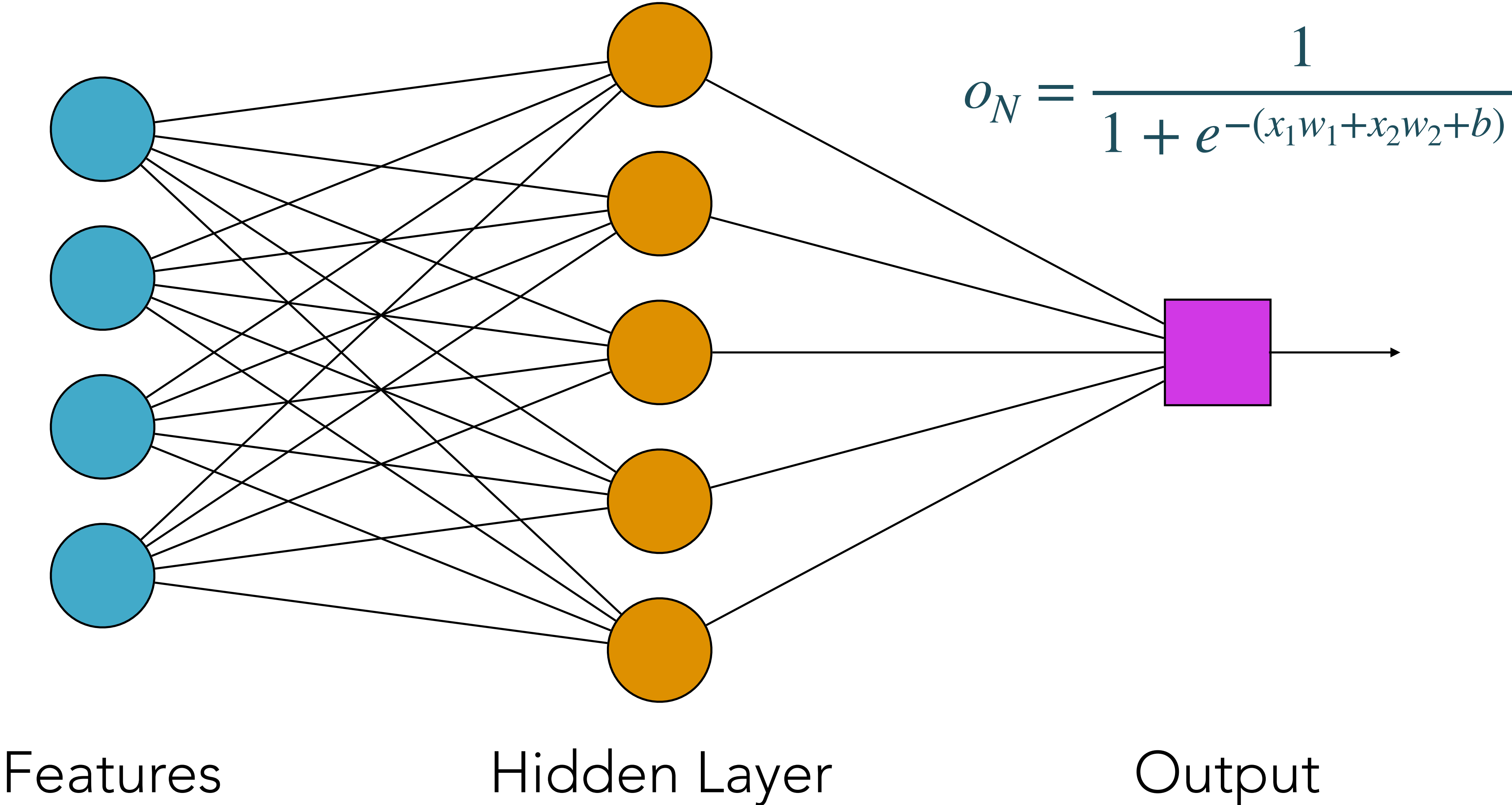


Features

Hidden Layer

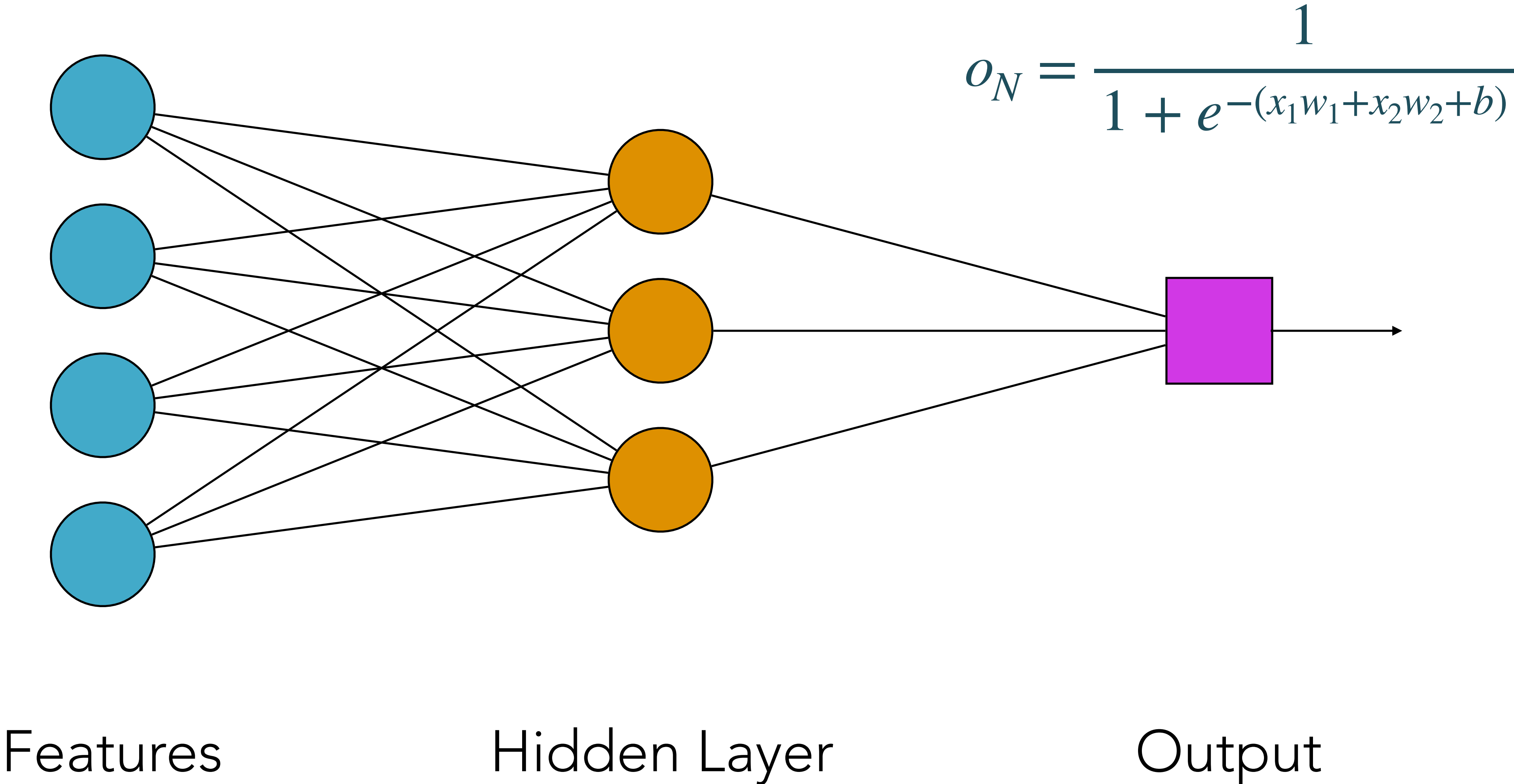
Output

# CHECK: HOW MANY TRAINABLE PARAMETERS?



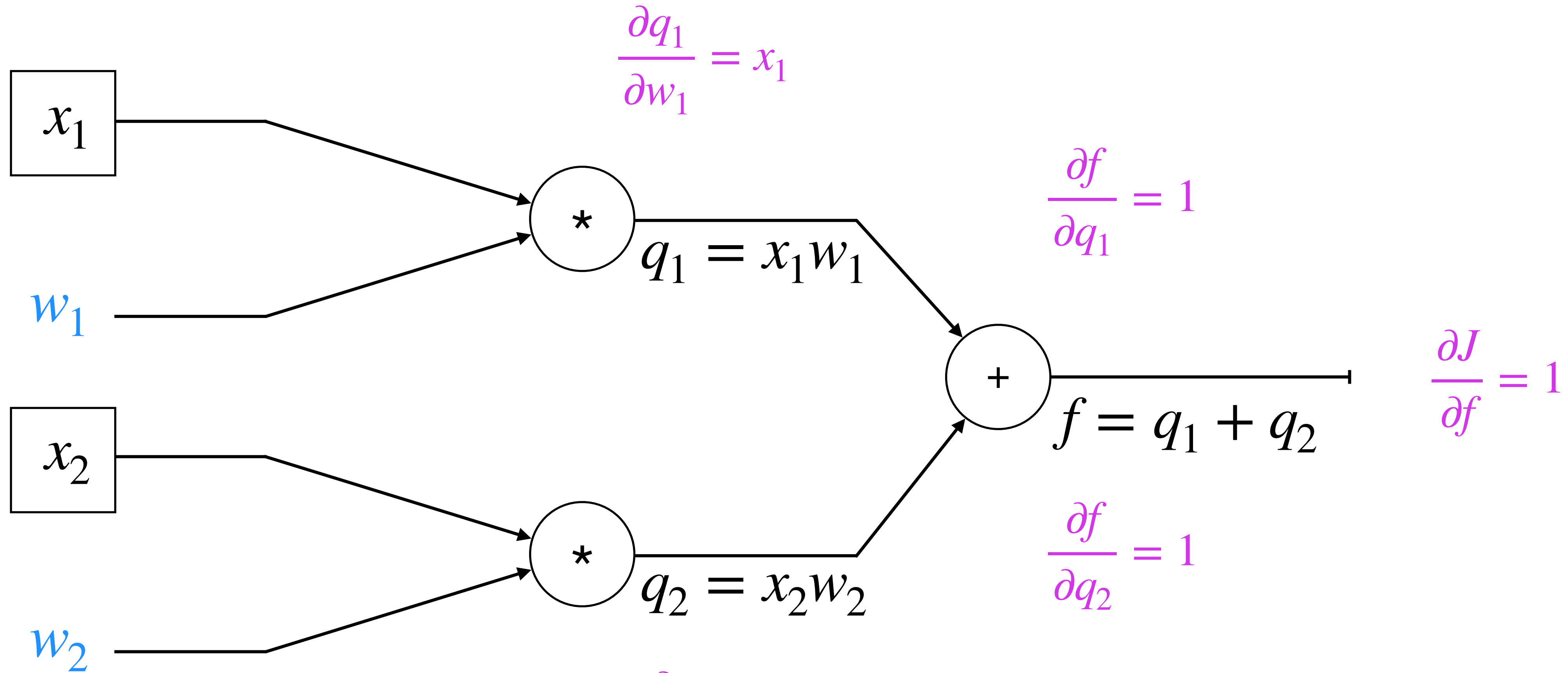


# CHECK: HOW MANY TRAINABLE PARAMETERS?



# BACKPROPAGATION

$$w_1 = w_1 + \eta * \frac{\partial J}{\partial f} \frac{\partial f}{\partial q_1} \frac{\partial q_1}{\partial w_1}$$

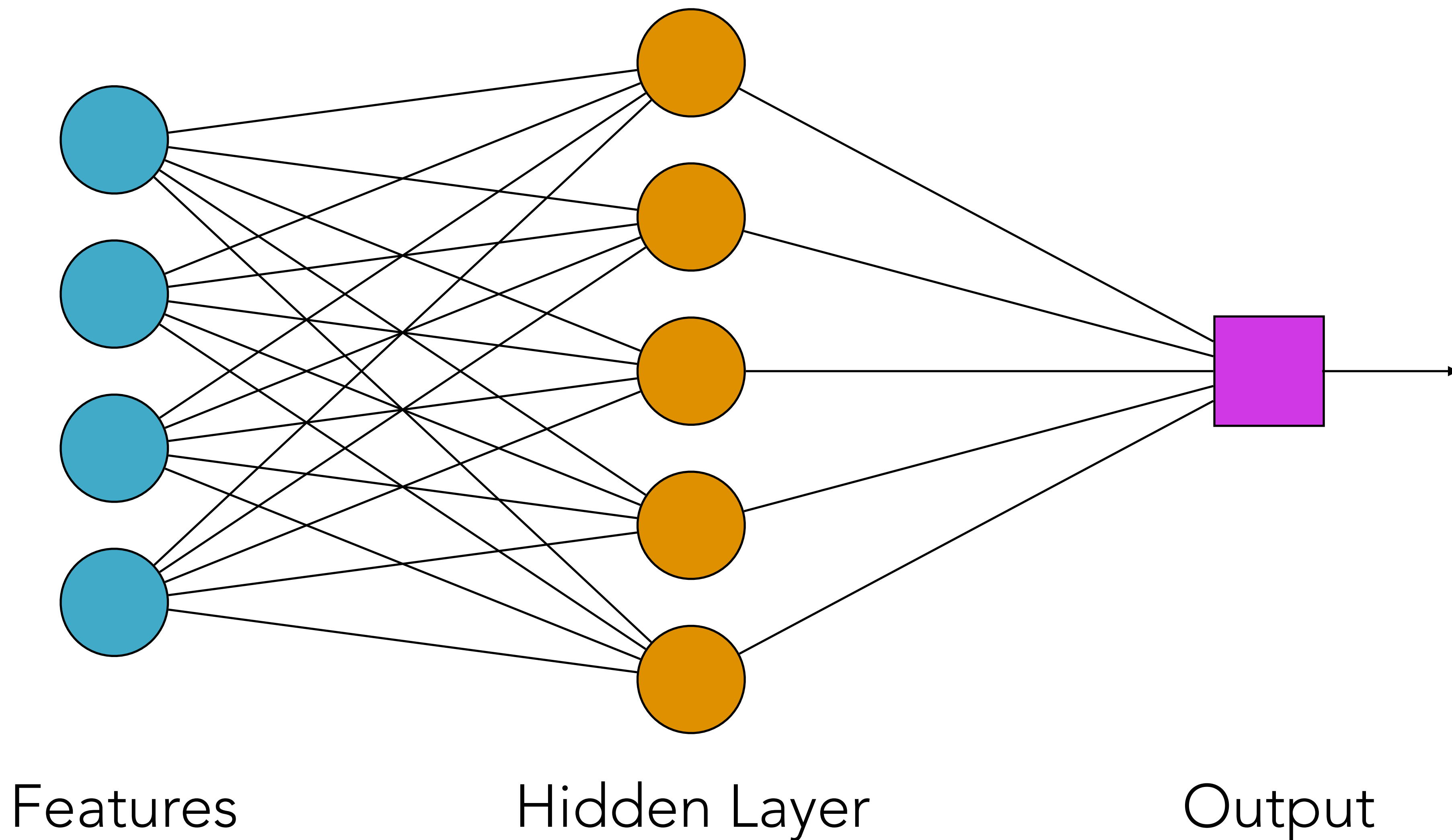


$$w_2 = w_2 + \eta * \frac{\partial J}{\partial f} \frac{\partial f}{\partial q_2} \frac{\partial q_2}{\partial w_2}$$

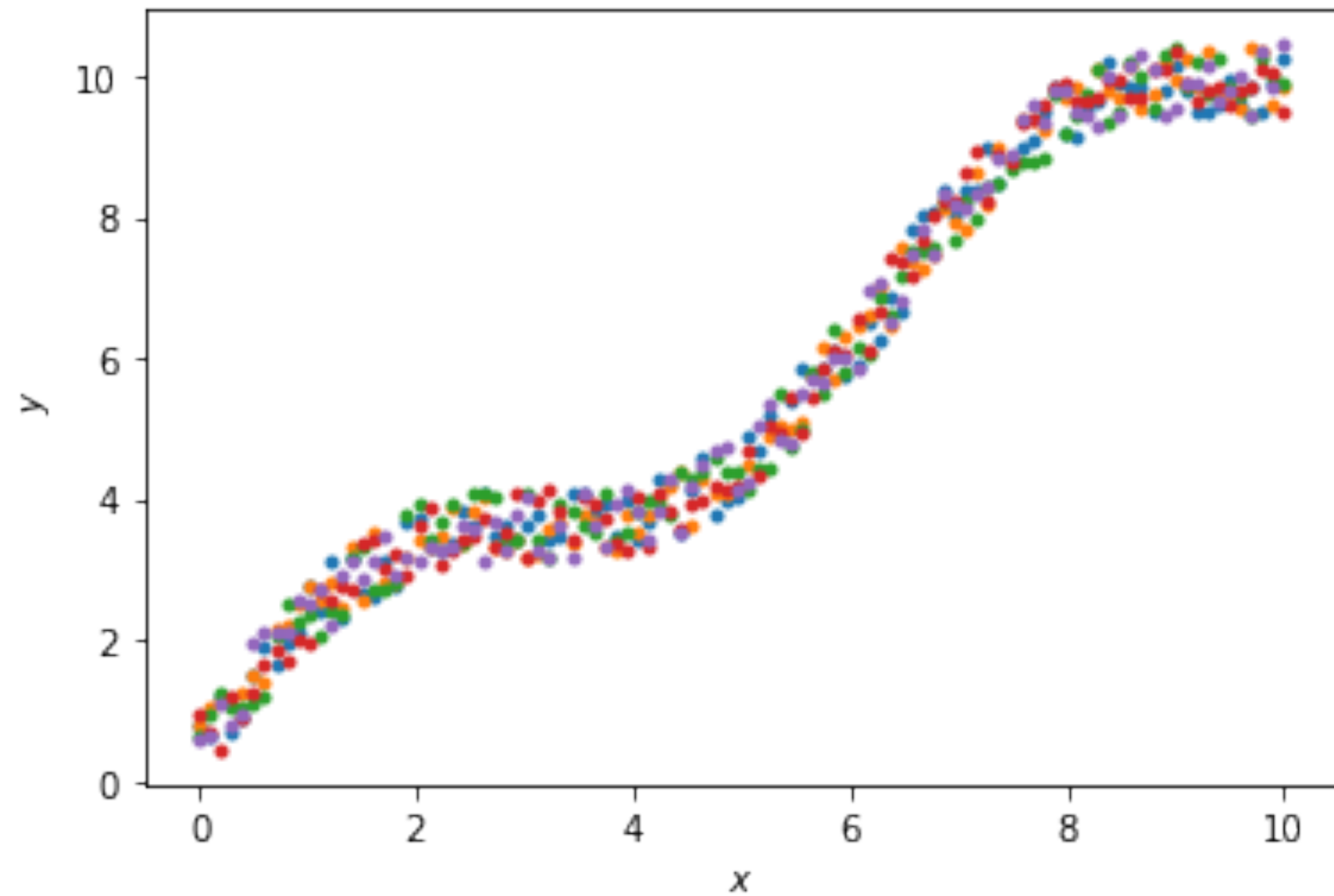
Loss function

$$J(w) = f - \hat{f}$$

**Weight initialization:** What happens if we initialize all weights to same value?



# LOSS FUNCTIONS



## Loss function

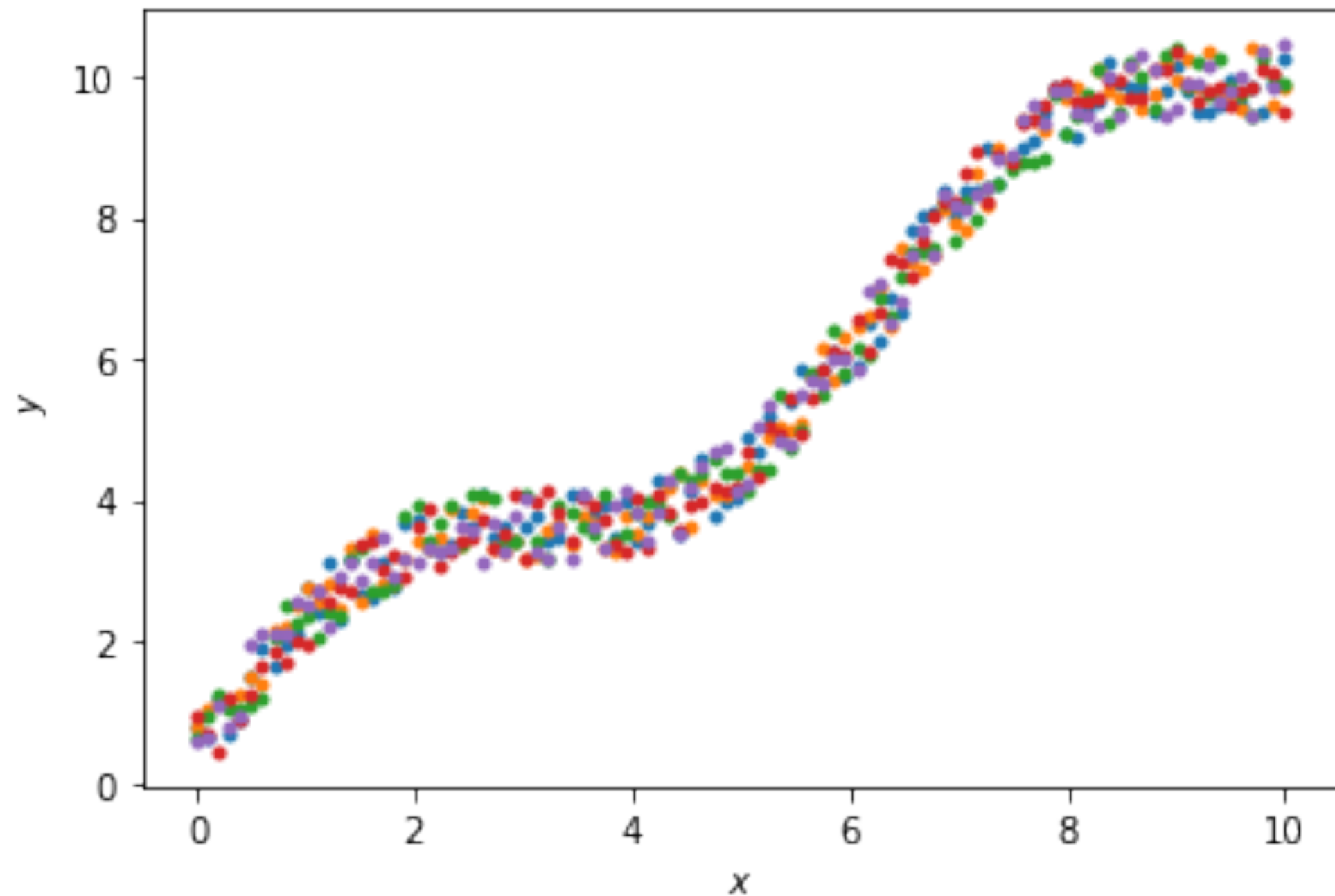
Mean squared error  
(MSE)

$$J(w) = \frac{1}{N} \sum_{i=0}^N (\hat{y}_i - y_i)^2$$

Mean absolute error  
(MAE)

$$J(w) = \frac{1}{N} \sum_{i=0}^N |\hat{y}_i - y_i|$$

# LOSS FUNCTIONS



## Loss function

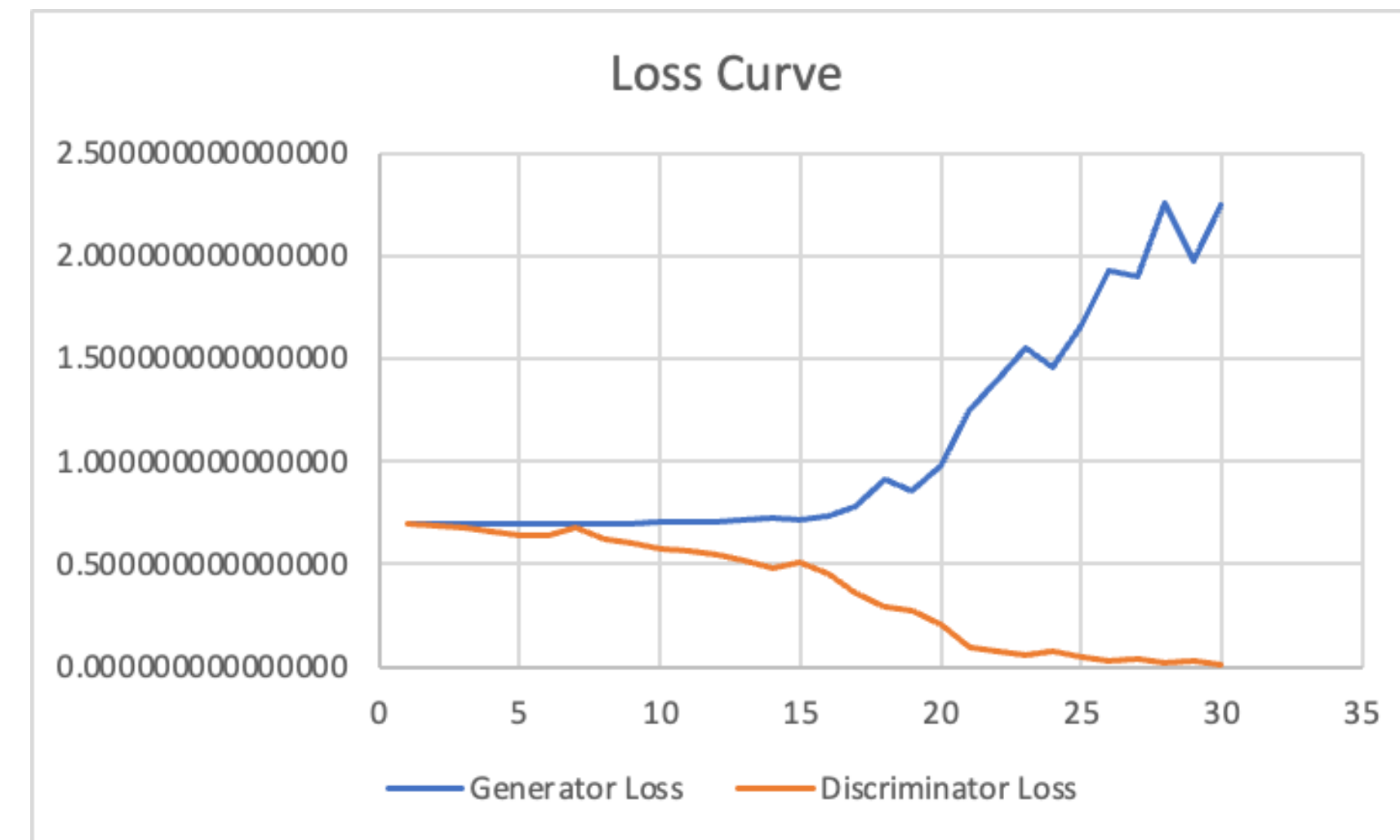
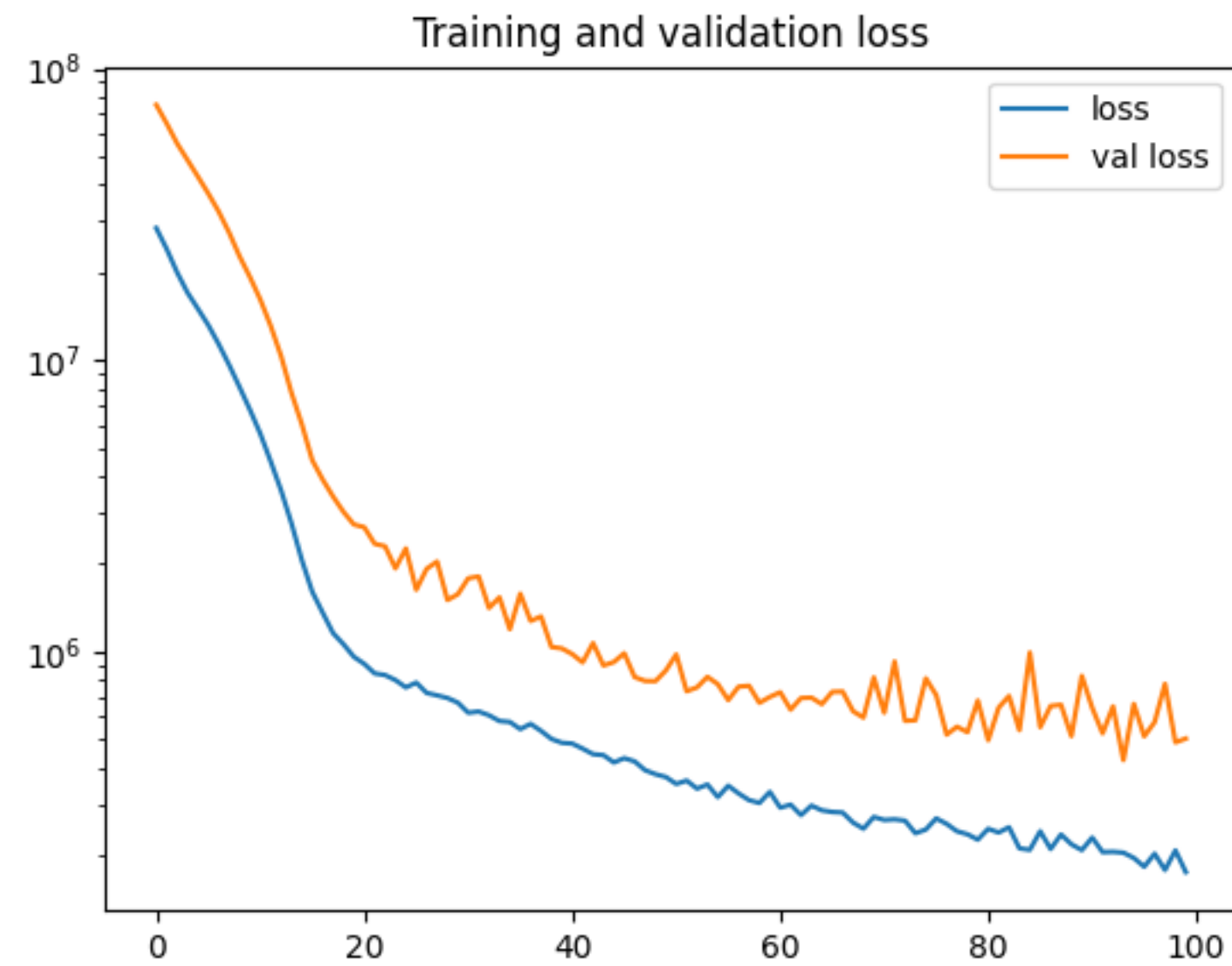
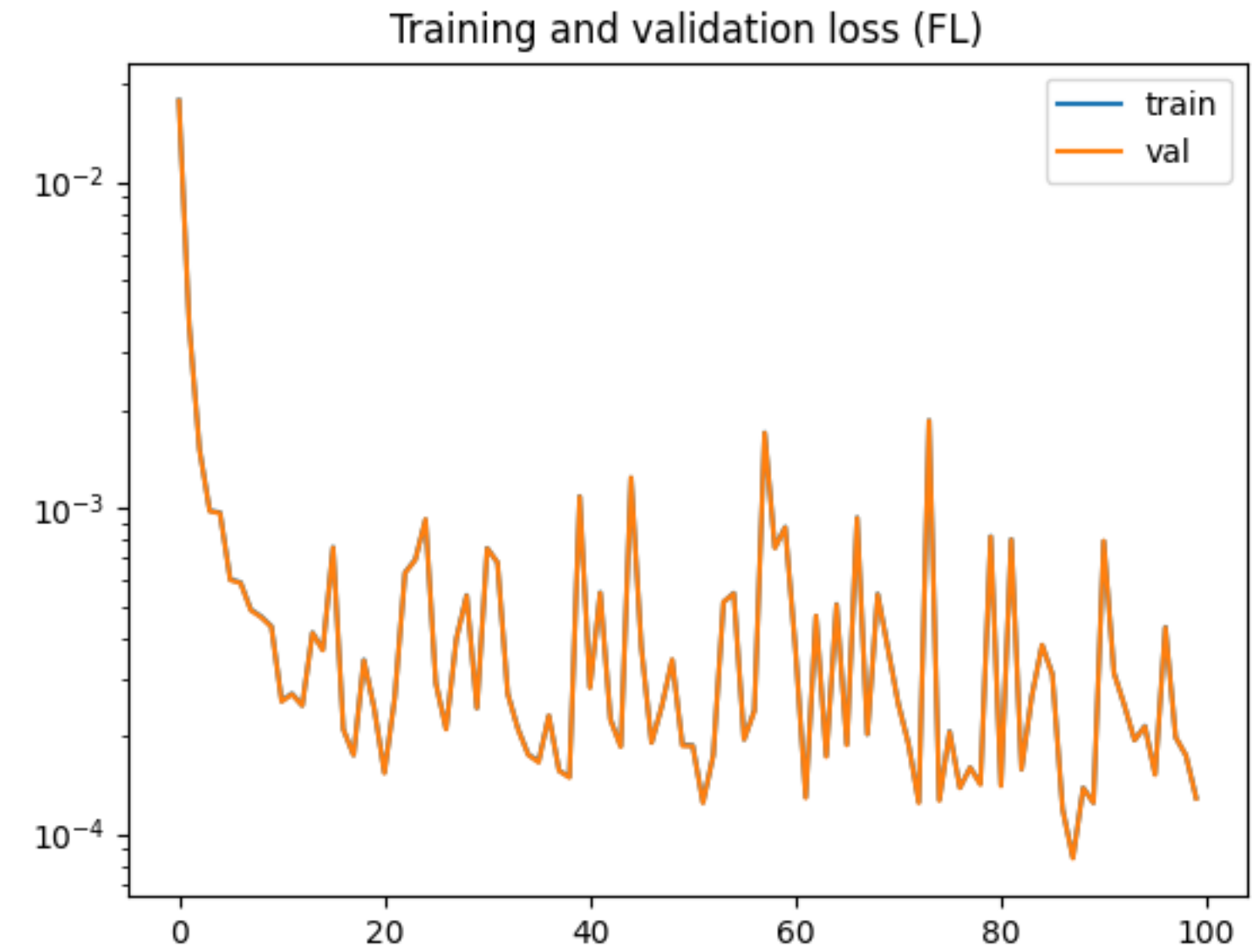
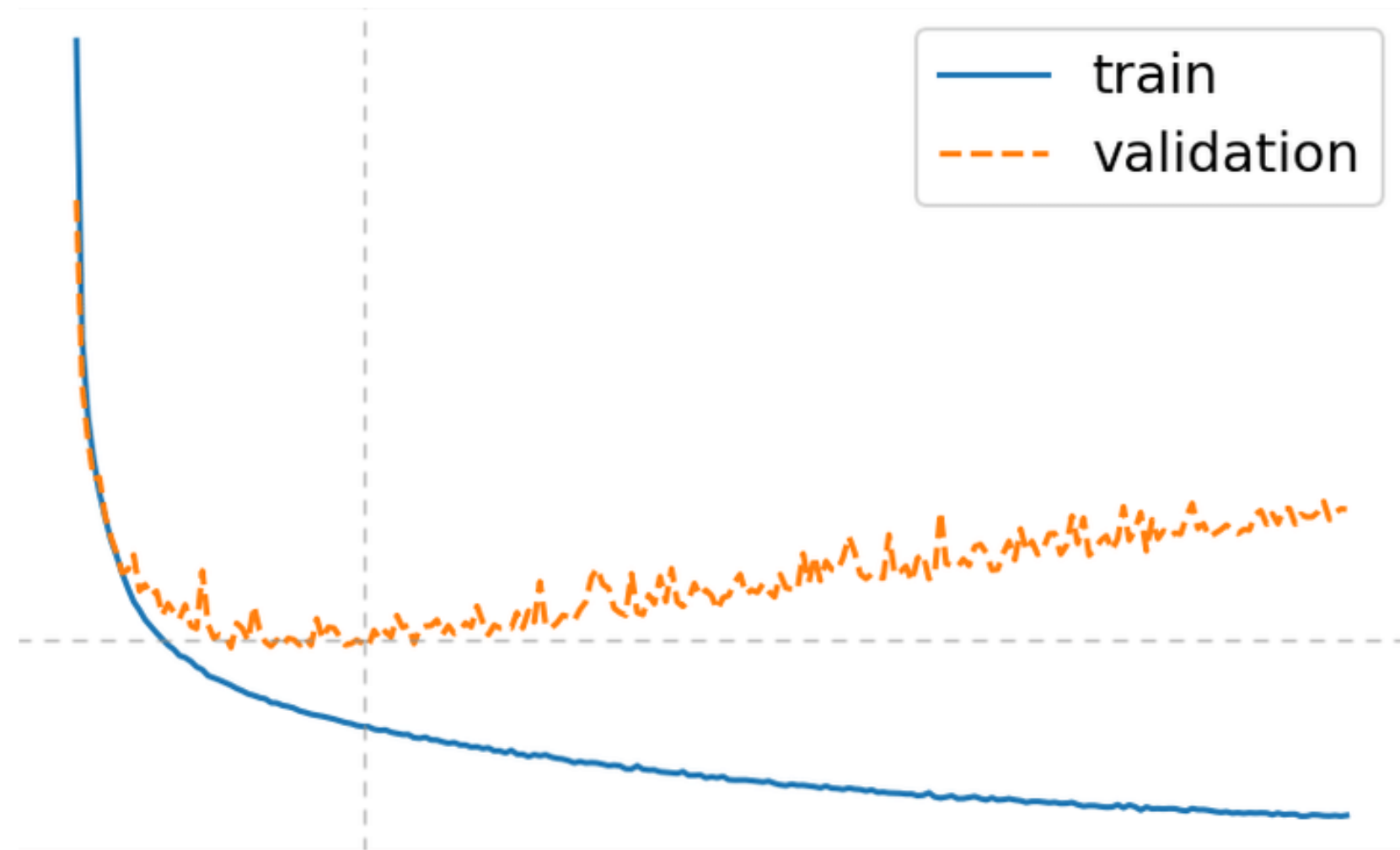
Mean squared error  
**mean**

$$J(w) = \frac{1}{N} \sum_{i=0}^N (\hat{y}_i - y_i)^2$$

Mean absolute error  
**median**

$$J(w) = \frac{1}{N} \sum_{i=0}^N |\hat{y}_i - y_i|$$

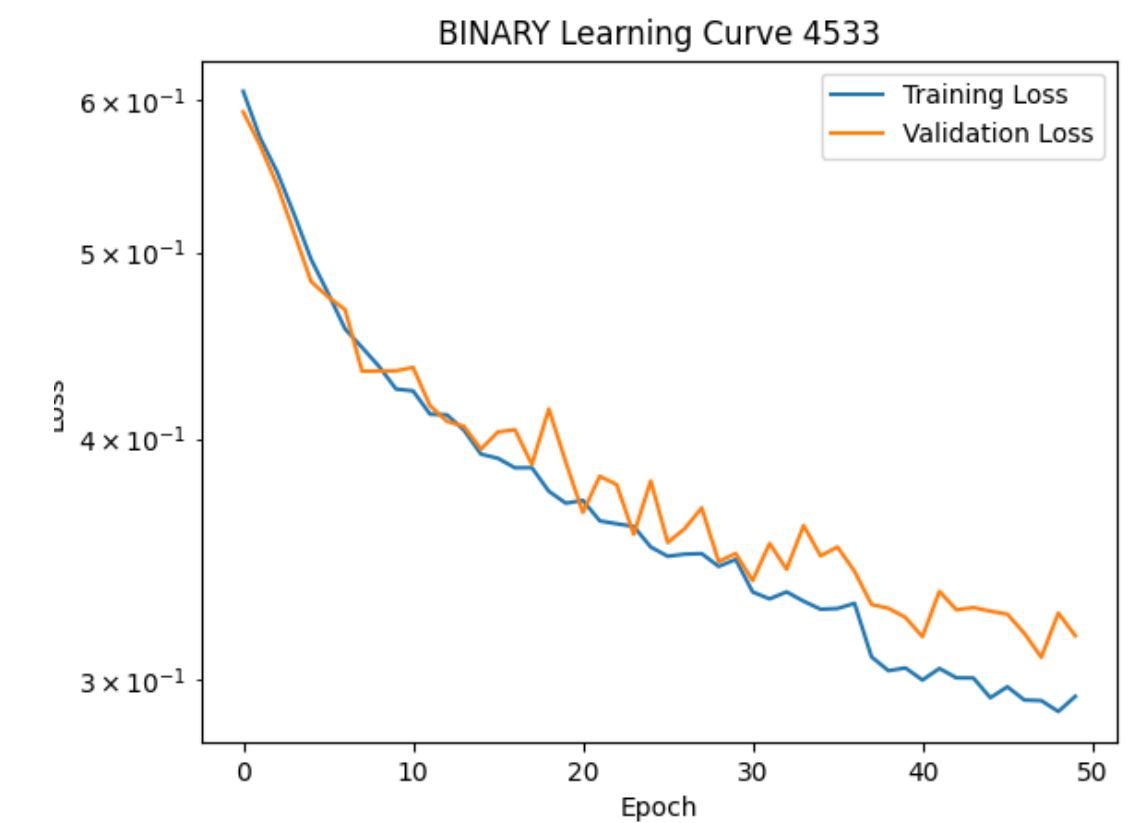
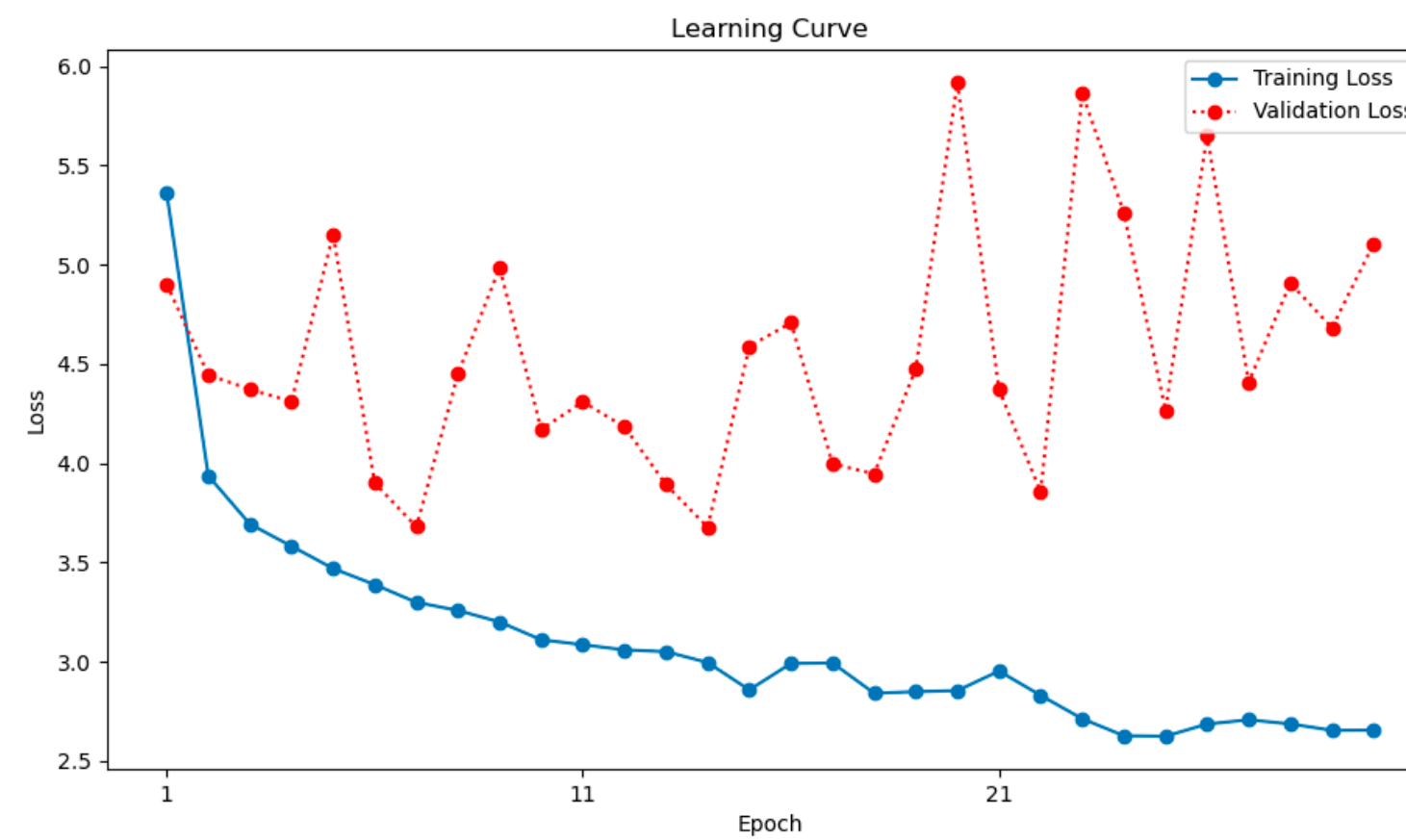
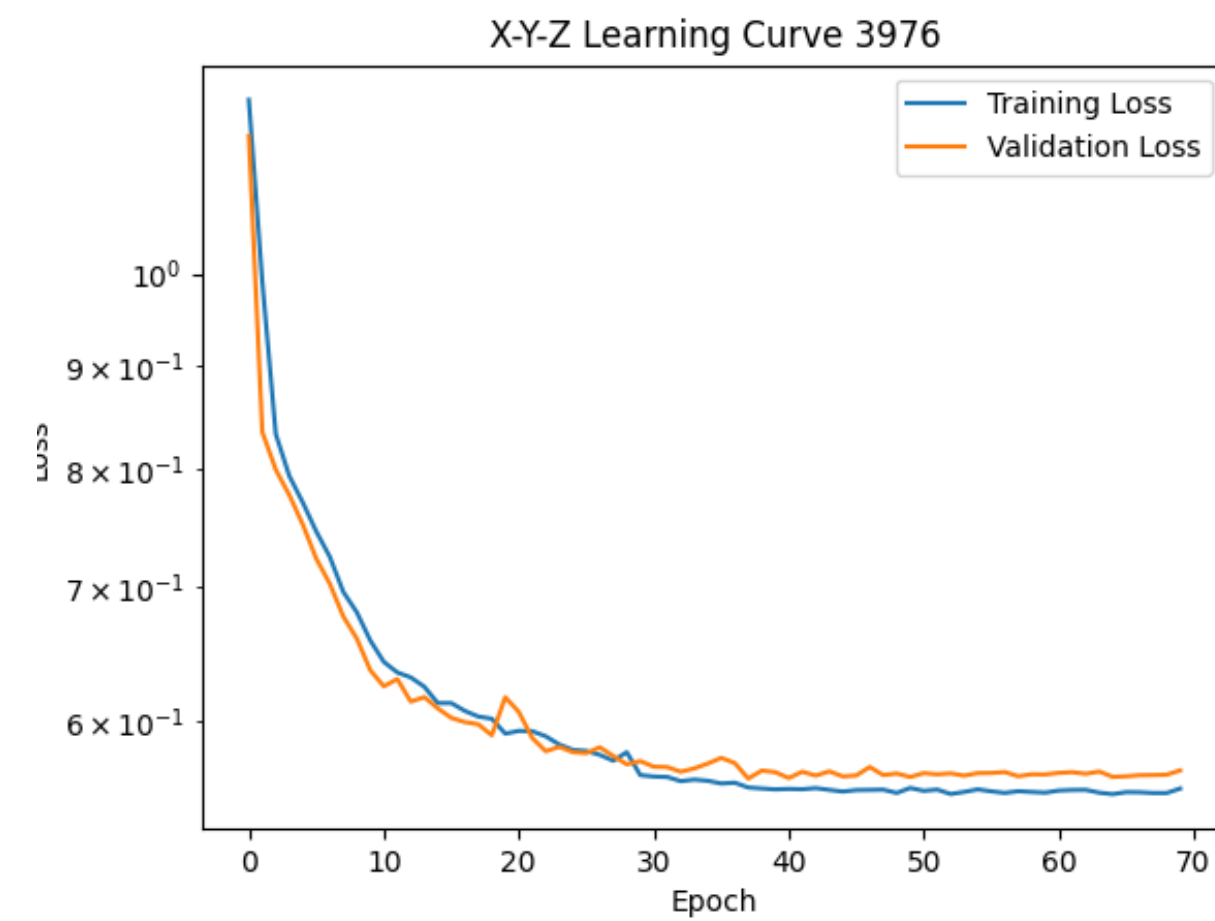
# Learning (loss) curves



# TRAINING

Remember that our goal is **NOT** to minimize loss on training data!

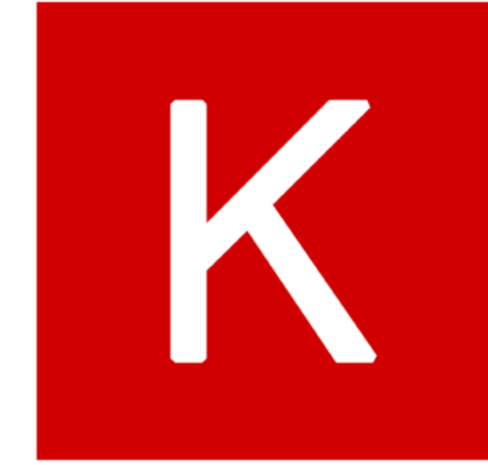
## Learning curves



# AUTOMATIC DIFFERENTIATION



TensorFlow



Keras



PyTorch

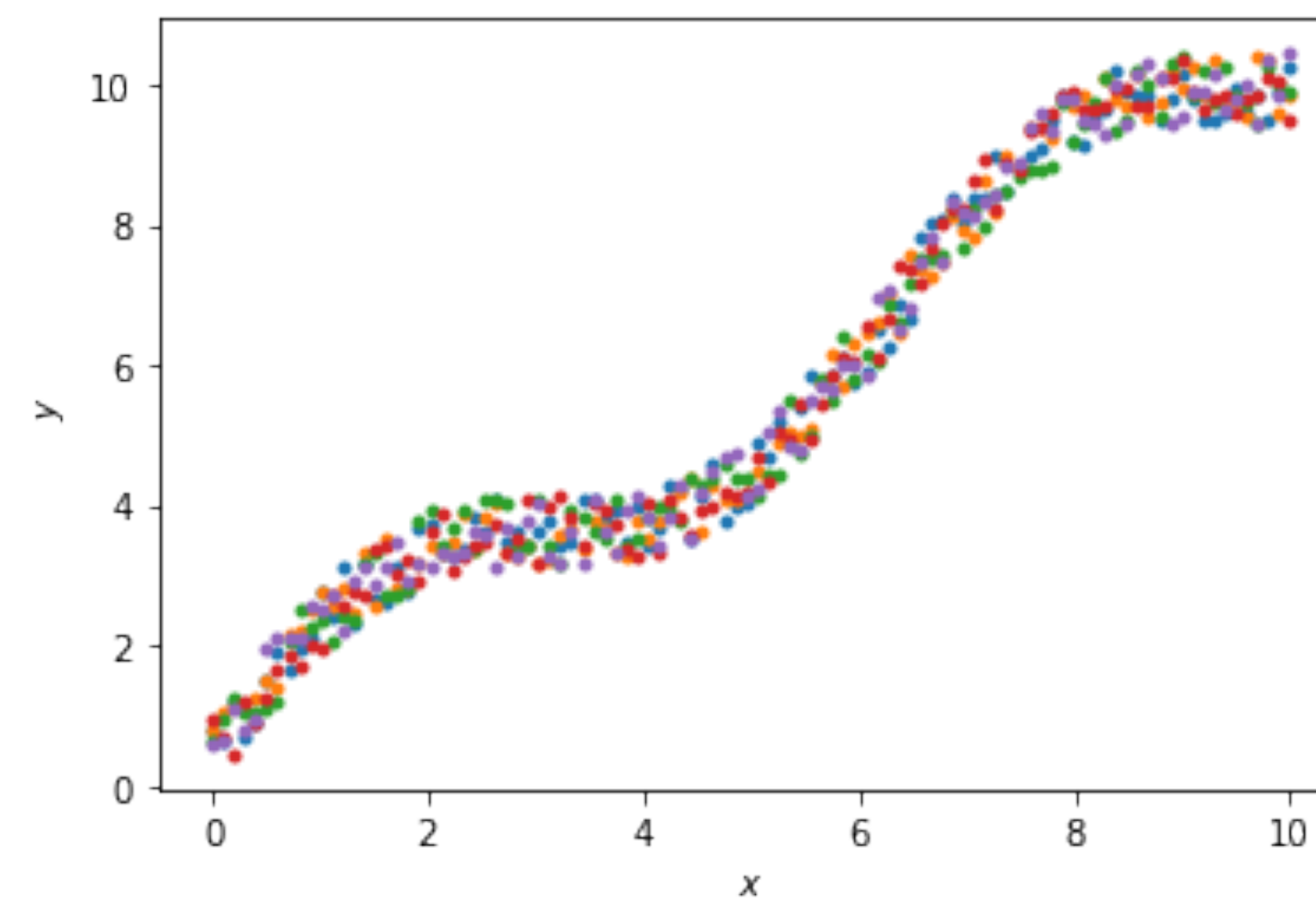




WHAT CAN WE DO WITH DEEP  
LEARNING?

# REGRESSION

	Featur e 1	Featur e 2	Featur e 3	Target 1	Target 2
Examp le 1					
Examp le 2					
Examp le 3					
Examp le 4					



# CLASSIFICATION

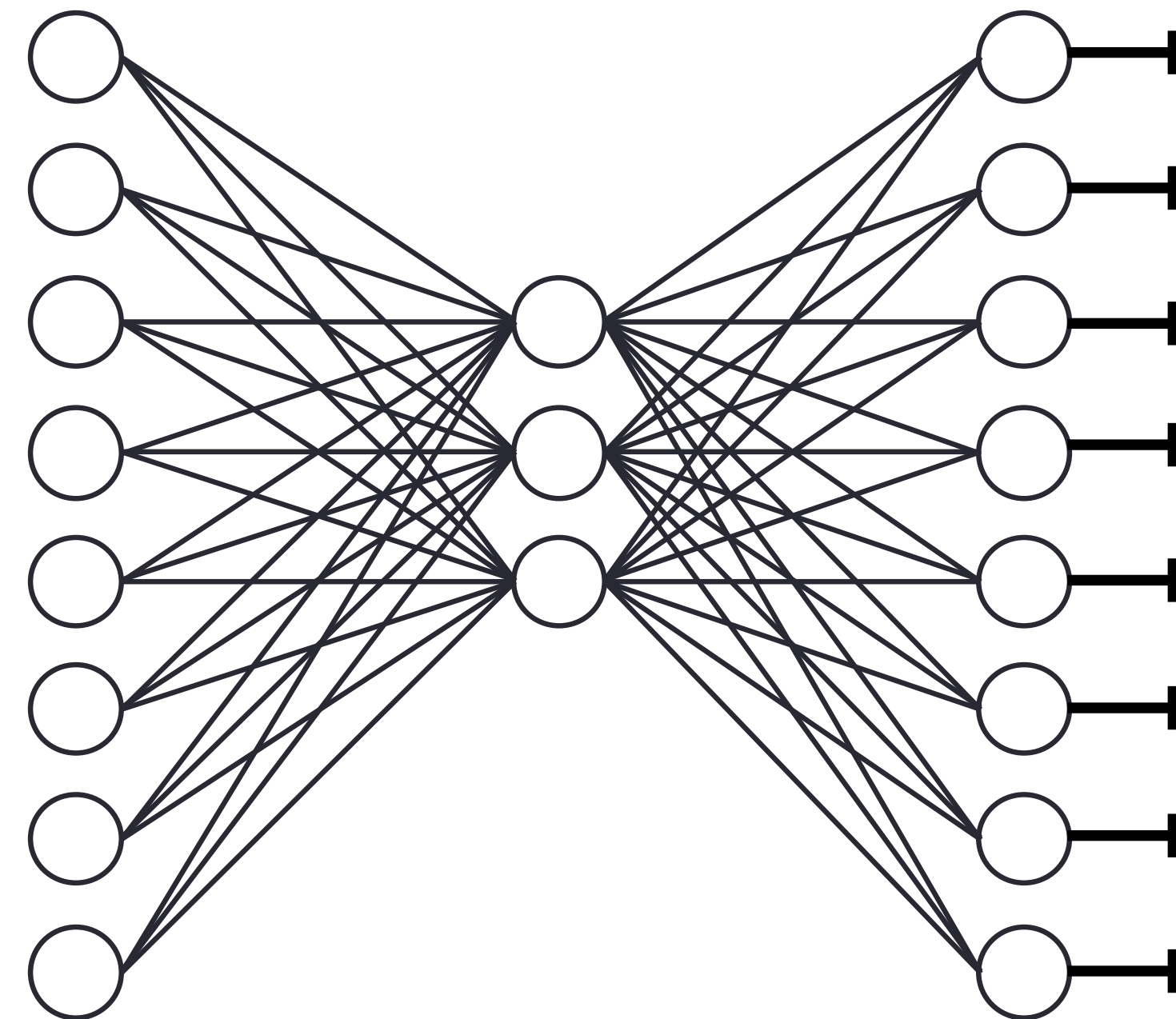
	Featur e 1	Featur e 2	Featur e 3	Target 1	Target 2
Exempl e 1					
Exempl e 2					
Exempl e 3					
Exempl e 4					

**Cross entropy:**

$$J = - (y \log(\hat{y}) + (1 - y)\log(1 - \hat{y}))$$

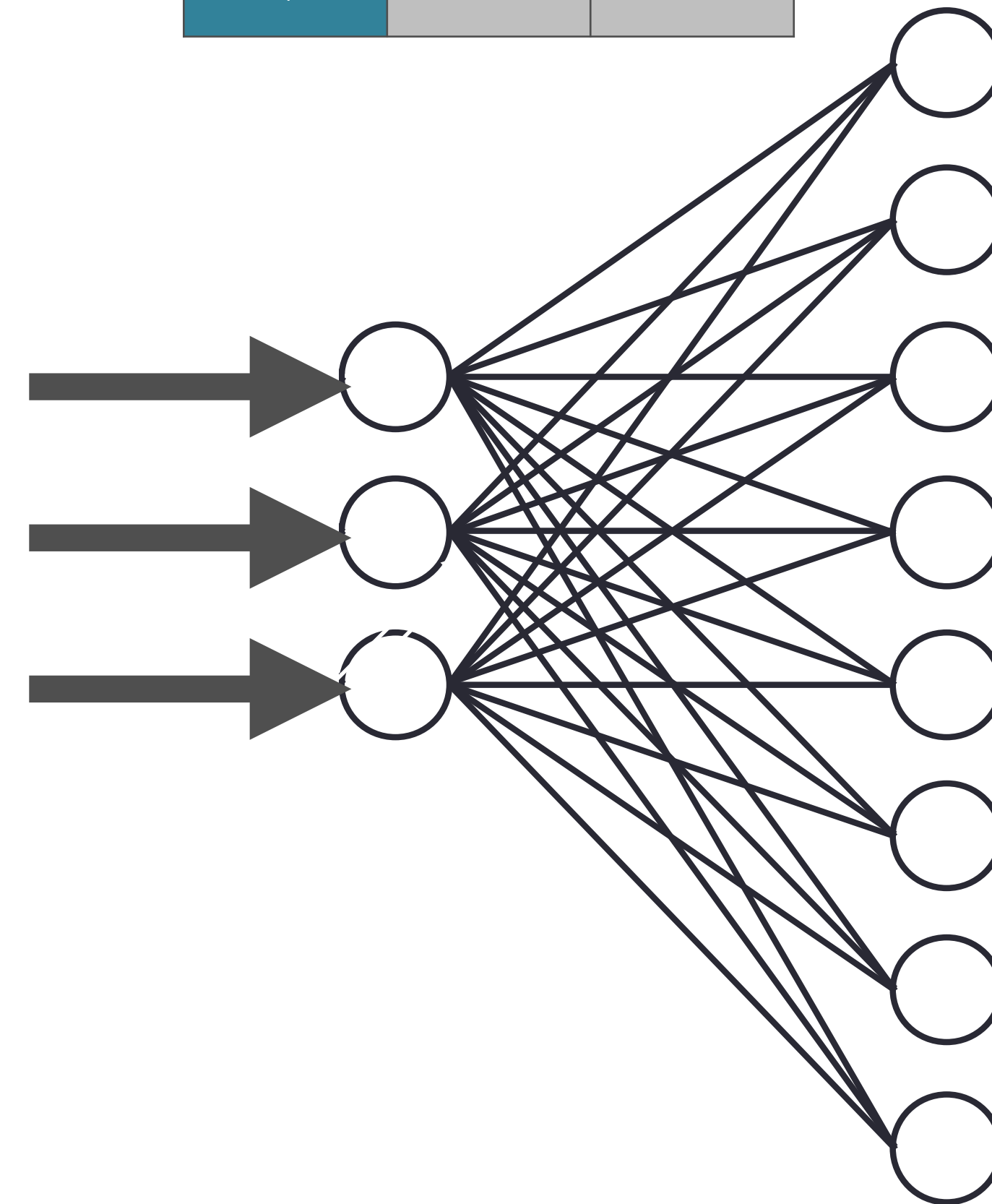
# SELF-SUPERVISED LEARNING

	Feature 1	Feature 2	Feature 3
Example 1			
Example 2			
Example 3			
Example 4			



# GENERATIVE MODELING

	Target 1	Target 2
Example 1		
Example 2		
Example 3		
Example 4		



BEST PRACTICES

LITERATURE

# CHOOSING AN ARCHITECTURE

HOW MANY LAYERS?

HOW MANY NODES PER LAYER?

LEARNING RATE

DROPOUT?

WHAT ACTIVATION FUNCTION(S)?

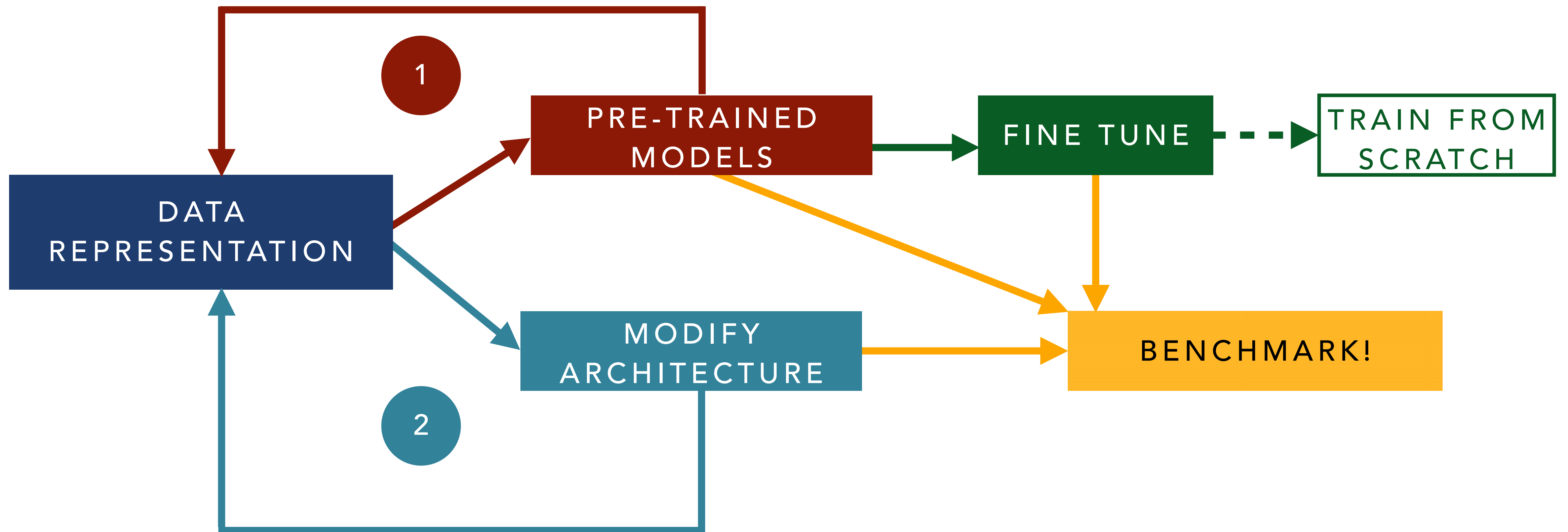
HOW MANY CONVOLUTION LAYERS?

FILTER SIZE?

STRIDE?

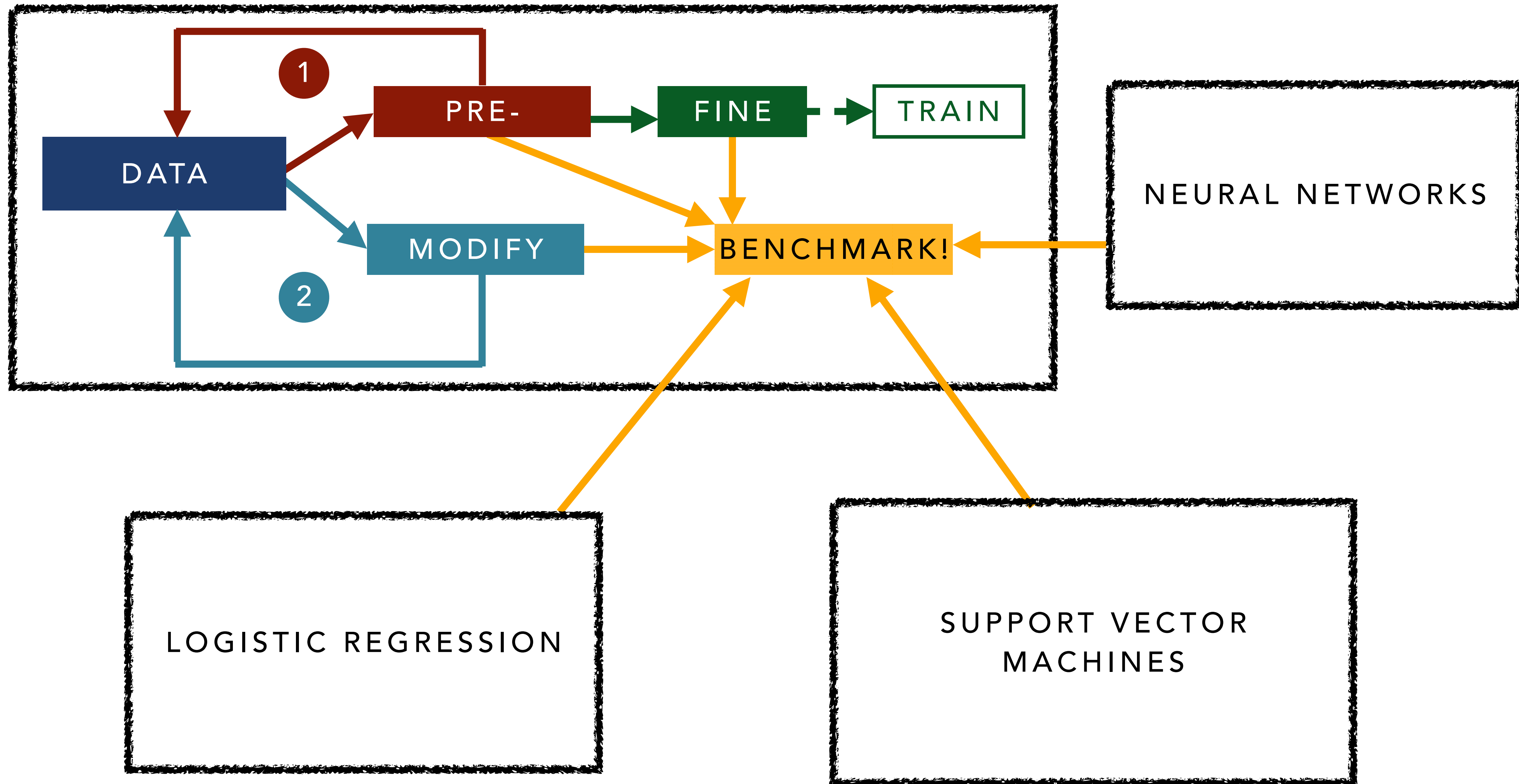
POOLING?

# EXAMPLE WORKFLOW





# EXAMPLE WORKFLOW



# MODEL CARDS

## Model Cards for Model Reporting

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, Timnit Gebru  
{mmitchellai, simonewu, andrewzaldivar, parkerbarnes, lucyvasserman, benhutch, espitzer, tgebru}@google.com  
deborah.raji@mail.utoronto.ca

### Model Card

- **Model Details.** Basic information about the model.
  - Person or organization developing model
  - Model date
  - Model version
  - Model type
  - Information about training algorithms, parameters, fairness constraints or other applied approaches, and features
  - Paper or other resource for more information
  - Citation details
  - License
  - Where to send questions or comments about the model
- **Intended Use.** Use cases that were envisioned during development.
  - Primary intended uses
  - Primary intended users
  - Out-of-scope use cases
- **Factors.** Factors could include demographic or phenotypic groups, environmental conditions, technical attributes, or others listed in Section 4.3.
  - Relevant factors
  - Evaluation factors
- **Metrics.** Metrics should be chosen to reflect potential real-world impacts of the model.
  - Model performance measures
  - Decision thresholds
  - Variation approaches
- **Evaluation Data.** Details on the dataset(s) used for the quantitative analyses in the card.
  - Datasets
  - Motivation
  - Preprocessing
- **Training Data.** May not be possible to provide in practice. When possible, this section should mirror Evaluation Data. If such detail is not possible, minimal allowable information should be provided here, such as details of the distribution over various factors in the training datasets.
- **Quantitative Analyses**
  - Unitary results
  - Intersectional results
- **Ethical Considerations**
- **Caveats and Recommendations**

# MODEL CARDS

## Model Cards for Model Reporting

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, Timnit Gebru  
{mmitchellai,simonewu,andrewzaldivar,parkerbarnes,lucyvasserman,benhutch,espitzer,tgebru}@google.com  
deborah.raji@mail.utoronto.ca

### Model Card

- **Model Details.** Basic information about the model.
  - Person or organization developing model
  - Model date
  - Model version
  - Model type
  - Information about training algorithms, parameters, fairness constraints or other applied approaches, and features
  - Paper or other resource for more information
  - Citation details
  - License
  - Where to send questions or comments about the model
- **Intended Use.** Use cases that were envisioned during development.
  - Primary intended uses
  - Primary intended users
  - Out-of-scope use cases
- **Factors.** Factors could include demographic or phenotypic groups, environmental conditions, technical attributes, or others listed in Section 4.3.
  - Relevant factors
  - Evaluation factors

- **Metrics.** Metrics should be chosen to reflect potential real-world impacts of the model.
  - Model performance measures
  - Decision thresholds
  - Variation approaches
- **Evaluation Data.** Details on the dataset(s) used for the quantitative analyses in the card.
  - Datasets
  - Motivation
  - Preprocessing
- **Training Data.** May not be possible to provide in practice. When possible, this section should mirror Evaluation Data. If such detail is not possible, minimal allowable information should be provided here, such as details of the distribution over various factors in the training datasets.
- **Quantitative Analyses**
  - Unitary results
  - Intersectional results
- **Ethical Considerations**
- **Caveats and Recommendations**

# PUBLICATIONS

## **Reproducibility:**

Transparent

Robust

# NEW RESEARCH

## **Conference papers**

NeurIPS: Neural Information Processing Systems

ICML: International Conference on Machine Learning

IJCAI: International Joint Conference on Artificial Intelligence

FAccT: Fairness, Accountability, and Transparency

NeurIPS: Machine Learning and the Physical Sciences Workshop

# PRACTICAL TIPS FOR TRAINING MODELS

# DATA

	Feature 1	Feature 2	Feature 3	Target
Example 1				
Example 2				
Example 3				
Example 4				



## NORMALIZATION

- Puts each feature on same scale
- Allows default hyperparameters to be a good starting point
  - learning rate, initialization of weights, etc.
- Options depend on data distribution
  - Standardization: mean: 0 stdev: 1
  - Min-max: [0,1]

# DATA

	Feature 1	Feature 2	Feature 3	Target
Example 1				
Example 2				
Example 3				
Example 4				

## ENCODING

- Non-numeric data
- Class-based features:
  - One-hot encoding:  $2 \rightarrow [0 \ 1]$
  - When classes do not have sequential meaning:  cars vs dogs vs plants  months



# BUILDING AND TRAINING MODELS

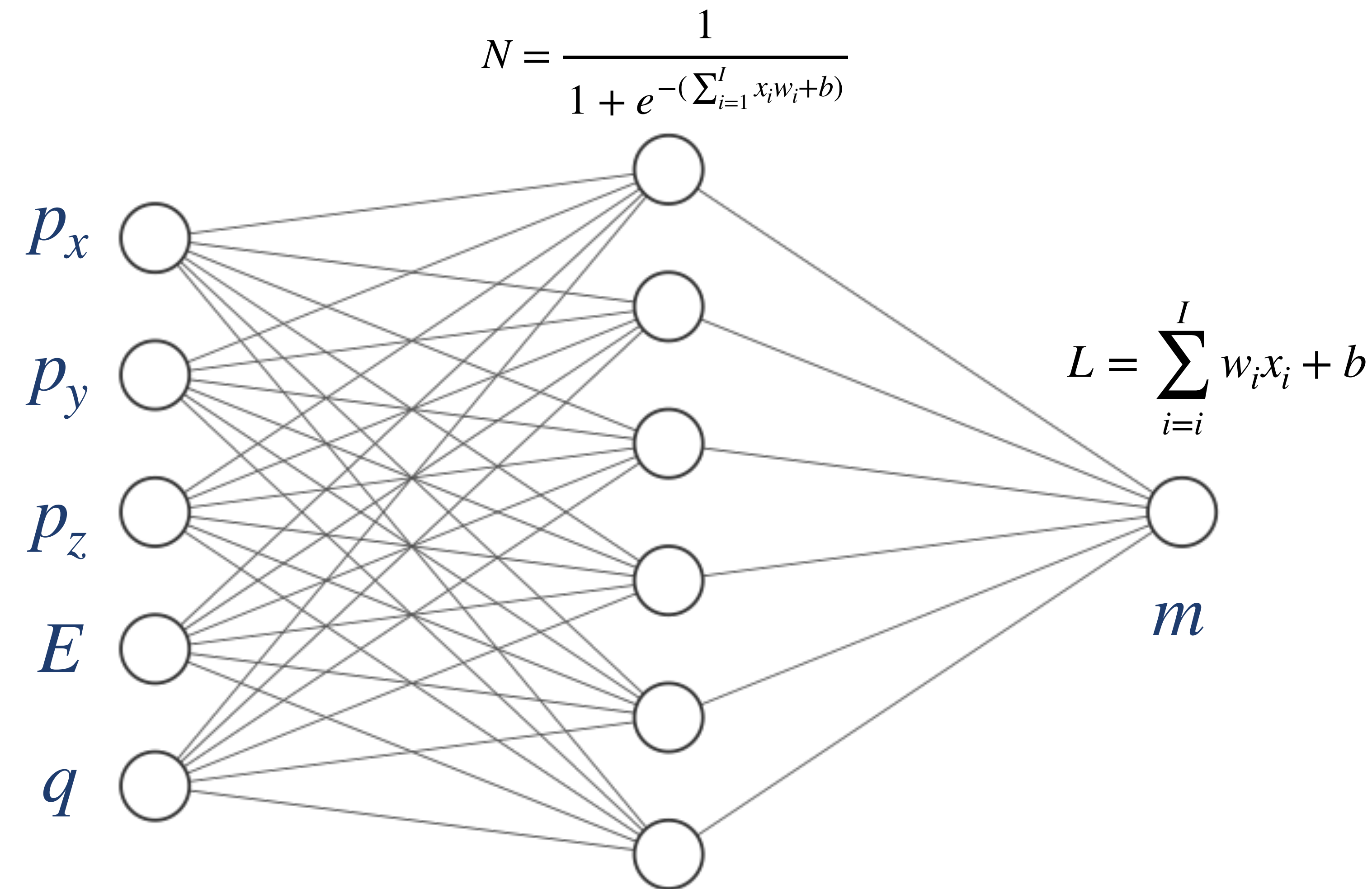
## TRAINING

- The most challenging part of machine learning is gaining the experience for tuning models well.
- We will work on this skill!

# ACTIVITY DESCRIPTION

- Simulating e+p collisions
- Predicting particle-level invariant mass (regression)
- Advanced: try a generative model (e.g. autoencoders)

# ACTIVITY DESCRIPTION



$$m^2 = E^2 - \|p\|^2$$

- Sigmoid activation for hidden layer and linear for output (regression model)
- How many "trainable parameters" in our model?

# COMMUNITY

- Each of you arrived here with your own backgrounds, specialty, and path in life
- Your experience and expertise are valuable here, no matter what it is
- If the activity is within your background, help others!
- If you are totally (or a little) lost, ask for help!
- It is our shared goal to have **each** of us leave with some new skill/knowledge/understanding

# GETTING STARTED

- Click the link under this tutorial on the workshop page
- If you have access to a google login, click "open in colab"
- Otherwise, download and open in Deepnote or download onto your personal computer (with appropriate dependencies)