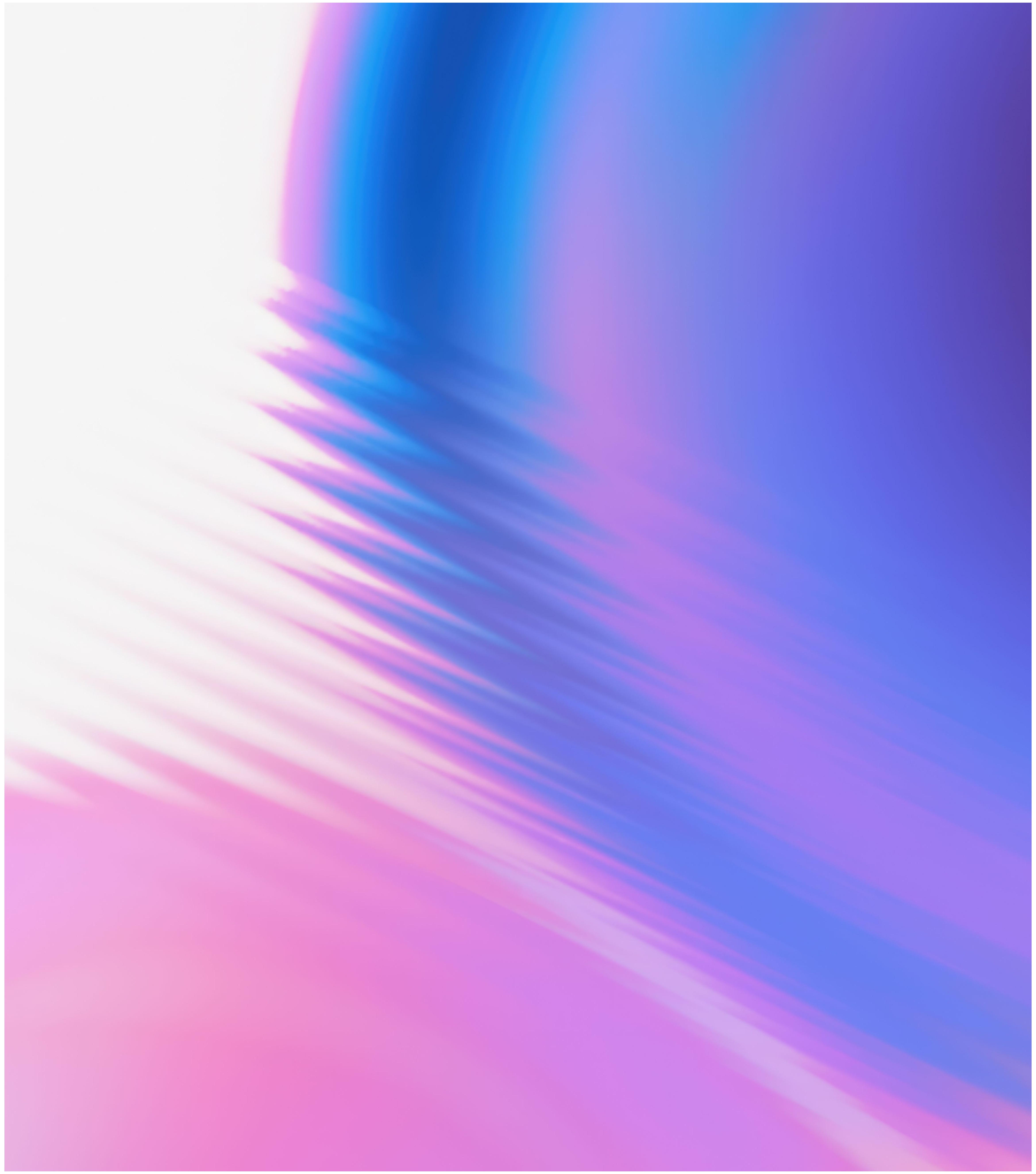
NTU Road to Utility Workshop 2025

Quantum Machine Learning

Boseong Kim
Workforce and Education
IBM Quantum

IBM



Part 1

Briefing quantum machine learning

Part 2

Quantum kernel estimation tutorial

Overview

1

Machine learning preliminaries

2

Variational circuits and data encoding

3

Quantum kernels and support vector machines

4

Quantum neural networks

Machine learning preliminaries

“Learning and adapting without following explicit instructions, by analyzing and drawing inferences from patterns in data”

Machine learning overview

Function approximation and optimization

$$g(x)$$

true function

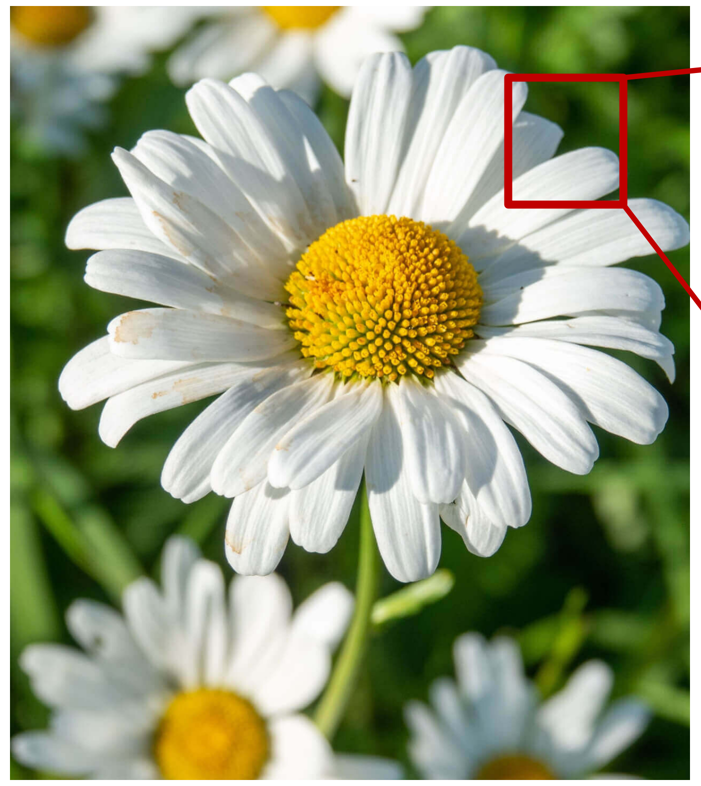
approximate $\longrightarrow$

$$f(\hat{x}, \vec{\theta})$$

mathematical model

x : data features

e.g. pixel values of an image



157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

e.g. $h(x, \vec{\theta}) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$

Goal: choose f

train $\vec{\theta}$

Machine learning types

1

Supervised Learning

- Classification
- Regression

2

Unsupervised Learning

- Dimensionality reduction
- Clustering
- Some generative models like GAN, autoencoder, etc.

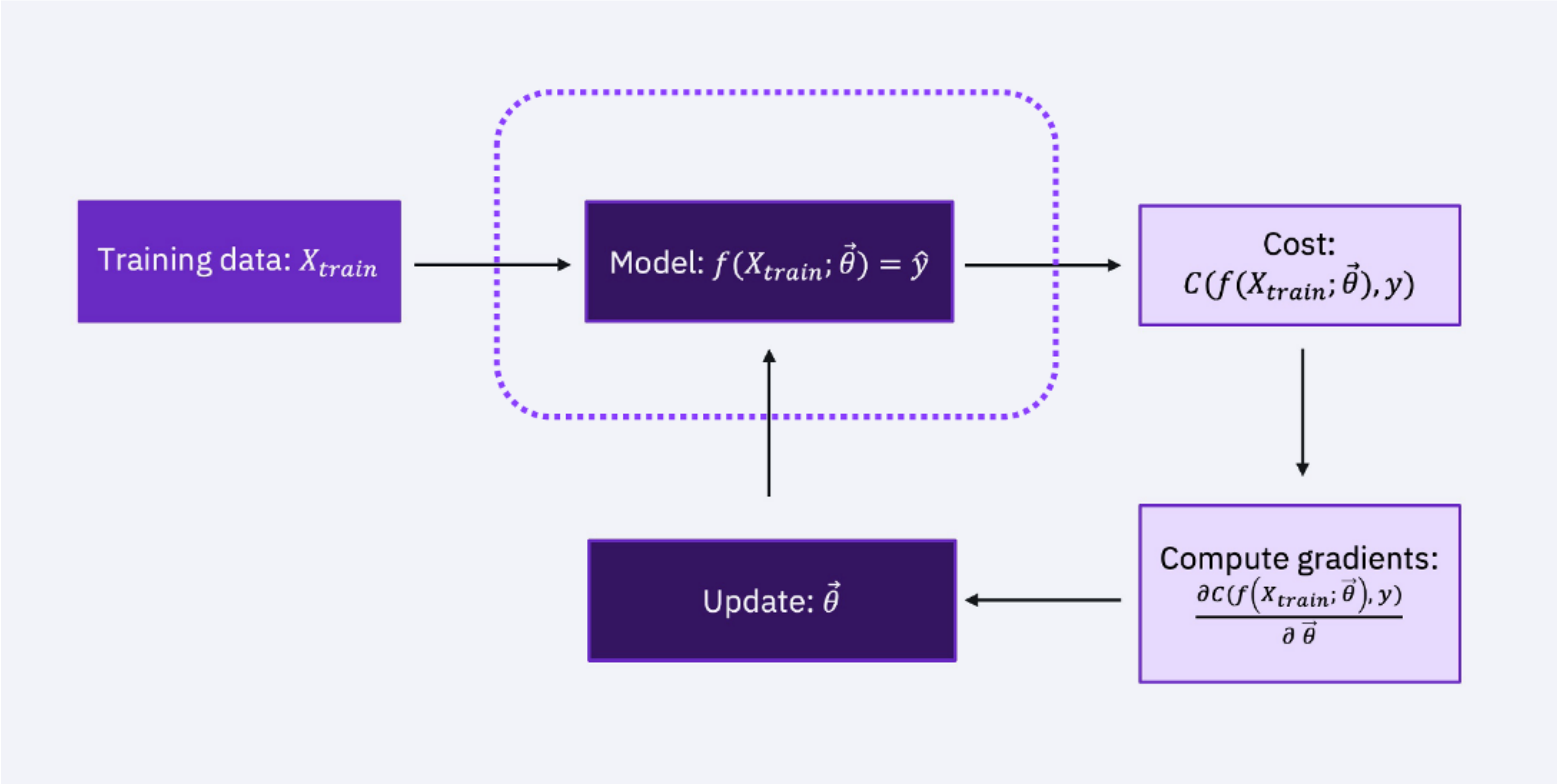
3

Reinforcement Learning

Agent maximizing rewards in an environment

Source: https://www.ibm.com/blog/machine-learning-types/

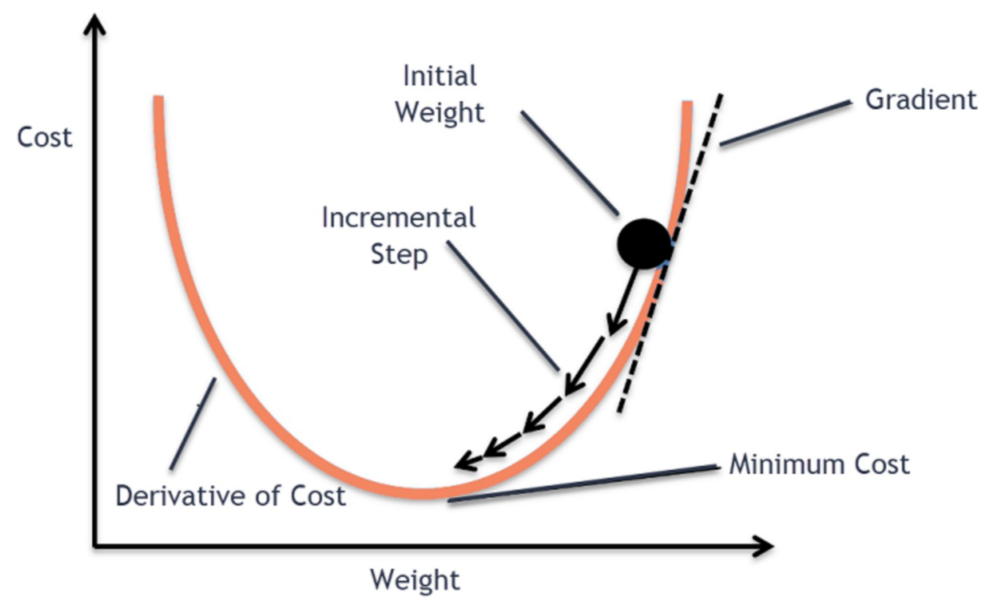
Supervised learning workflow



e.g. Mean squared error

$$\mathrm{MSE} = \frac{1}{n} \sum_{i=1}^{n} \left(Y_i - \hat{Y}_i \right)^2 .$$

e.g. Gradient descent

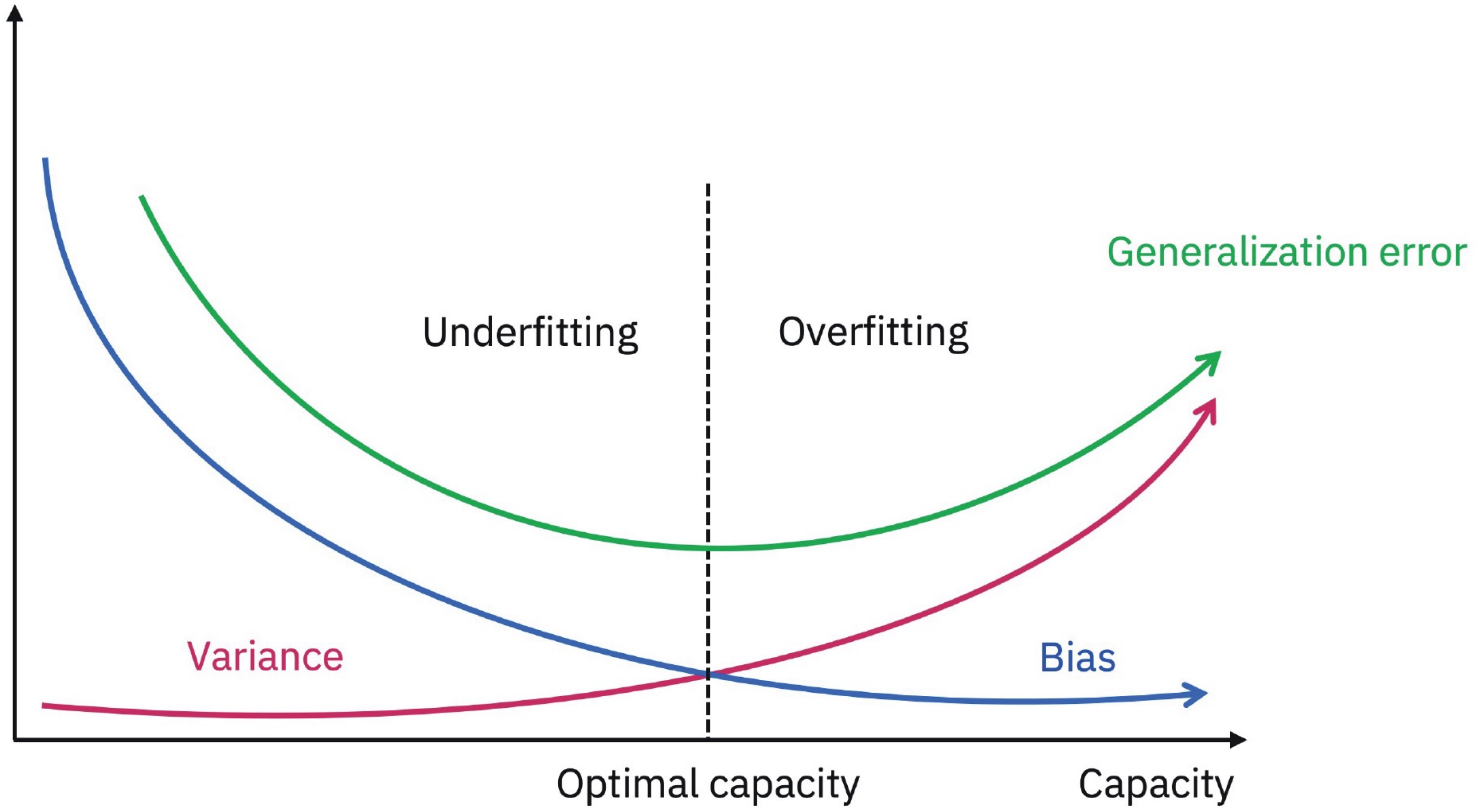


Model validation

Model should work well both on training and the test data

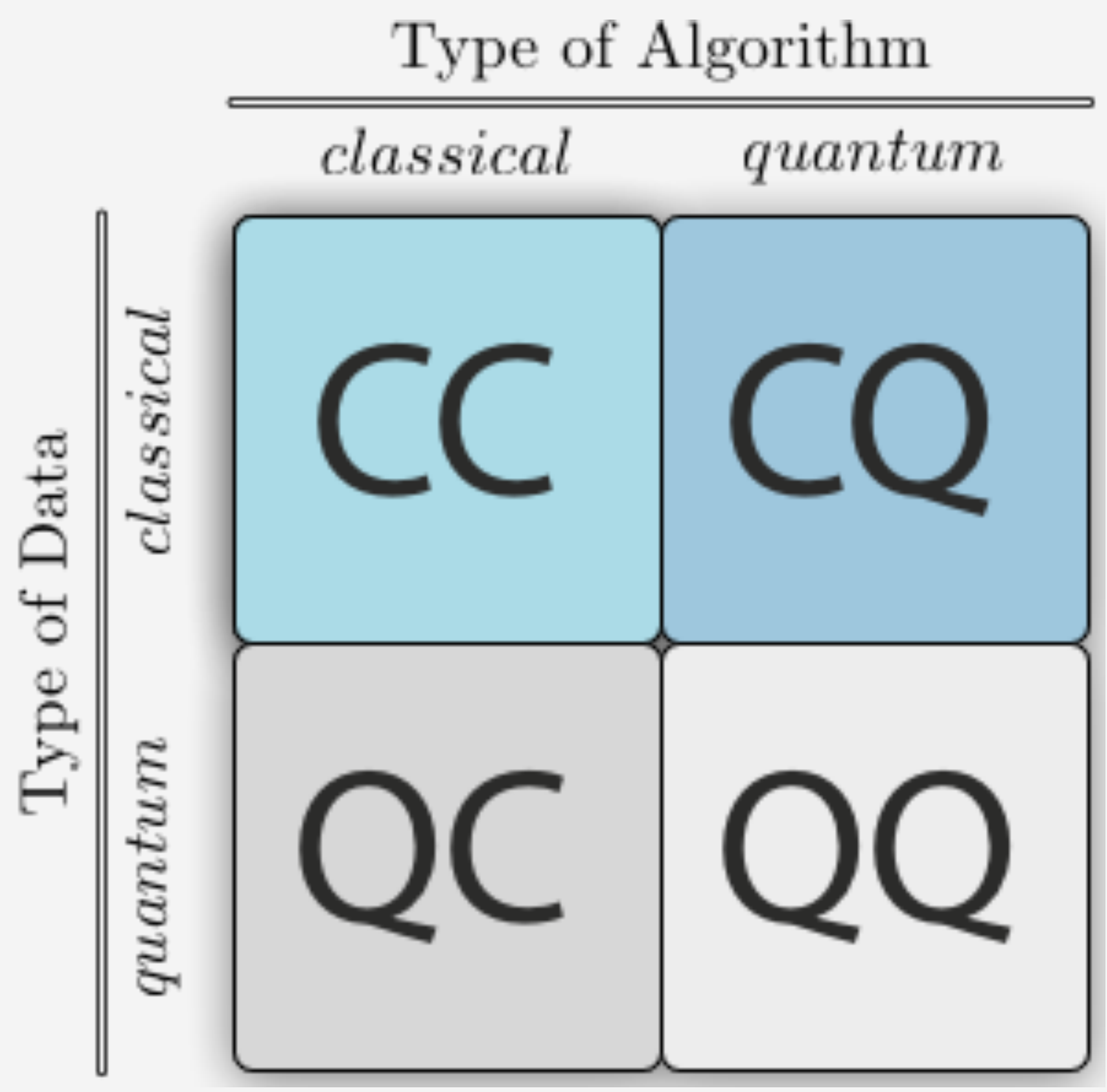
The model should not overfit or underfit to training data (poor generalization)

”bias-variance” trade-off

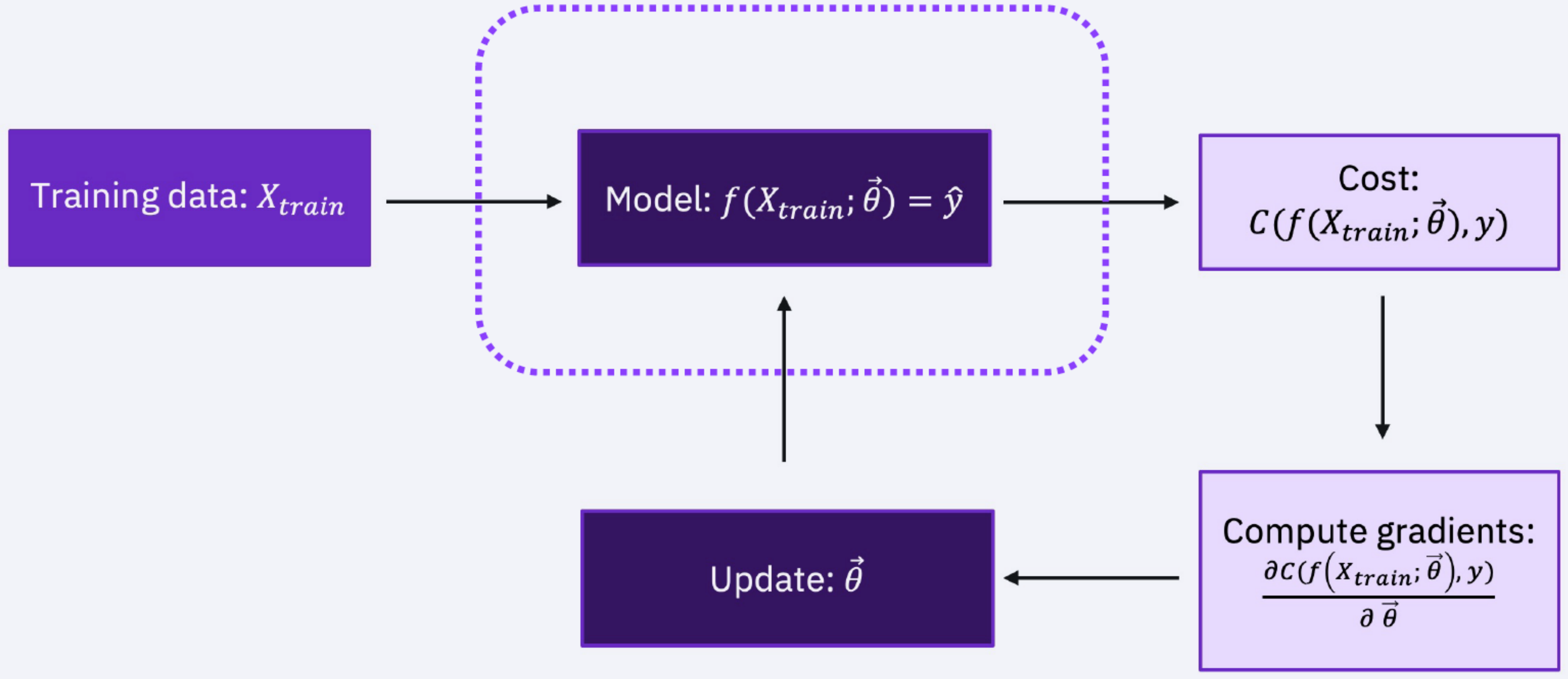


Variational circuits and data encoding

Quantum machine learning



Schuld, Maria, and Francesco Petruccione. Supervised learning with quantum computers. Vol. 17. Berlin: Springer, 2018.



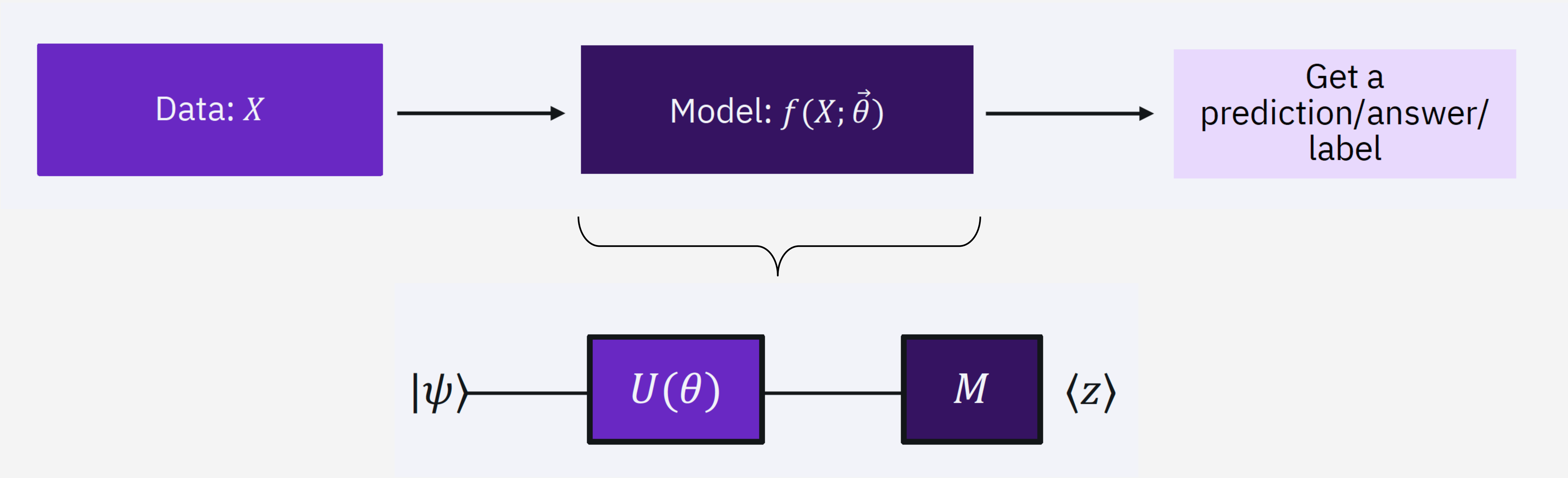
Also: **near-term** vs fault-tolerant

- Quantum SVM
- Quantum NNs

- HHL algorithm
- Quantum PCA

Harrow, Aram W., Avinatan Hassidim, and Seth Lloyd. "Quantum algorithm for linear systems of equations." Physical review letters 103.15 (2009): 150502.
Lloyd, Seth, Masoud Mohseni, and Patrick Rebentrost. "Quantum principal component analysis." Nature Physics 10.9 (2014): 631-633.

Variational circuit as a classifier



Variational circuit

Parametrized quantum circuit (PQC)

Ansatz

Variational circuit as a classifier

Task: Supervised learning (suppose binary classification, {1, -1})

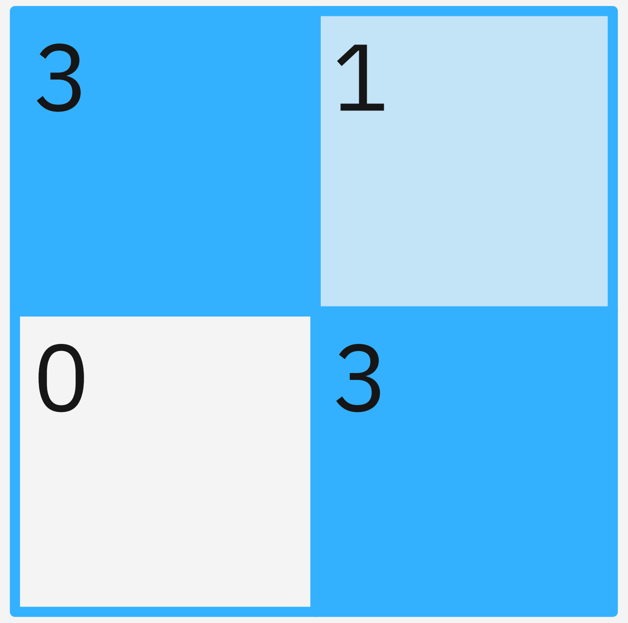
Step 1: Encode the classical data into a quantum state

Step 2: Apply a parameterized model

Step 3: Measure the circuit to extract labels

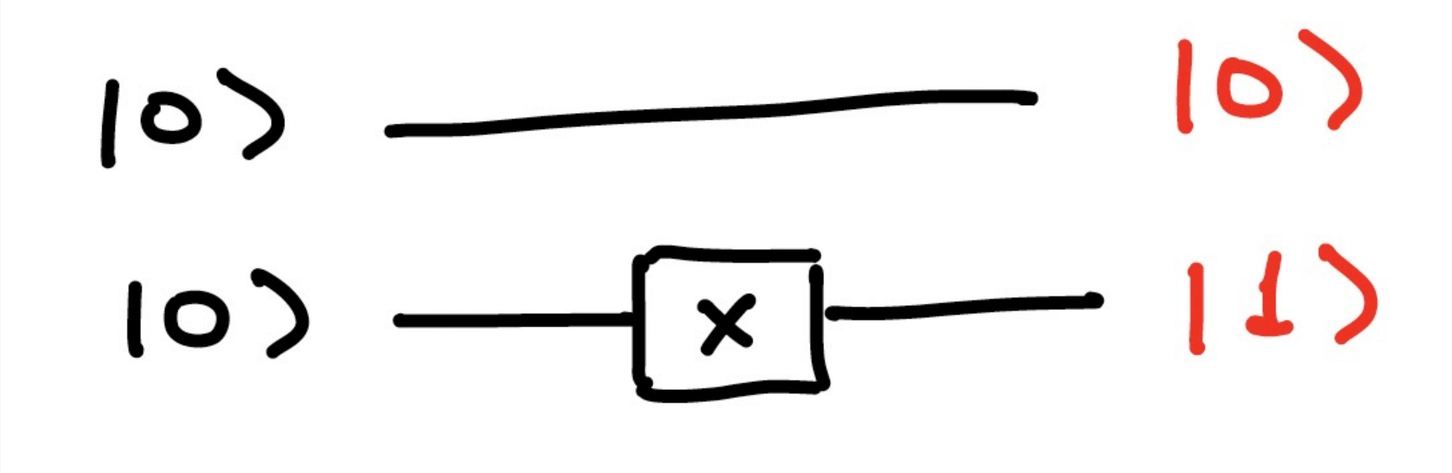
Step 4: Use optimization techniques (like gradient descent) to update model parameters

Data encoding



Basis encoding: Encode each n-bit feature into n qubits

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 0 \\ 3 \end{bmatrix} = \begin{bmatrix} 11 \\ 01 \\ 00 \\ 11 \end{bmatrix} = \begin{bmatrix} |11\rangle \\ |01\rangle \\ |00\rangle \\ |11\rangle \end{bmatrix}$$



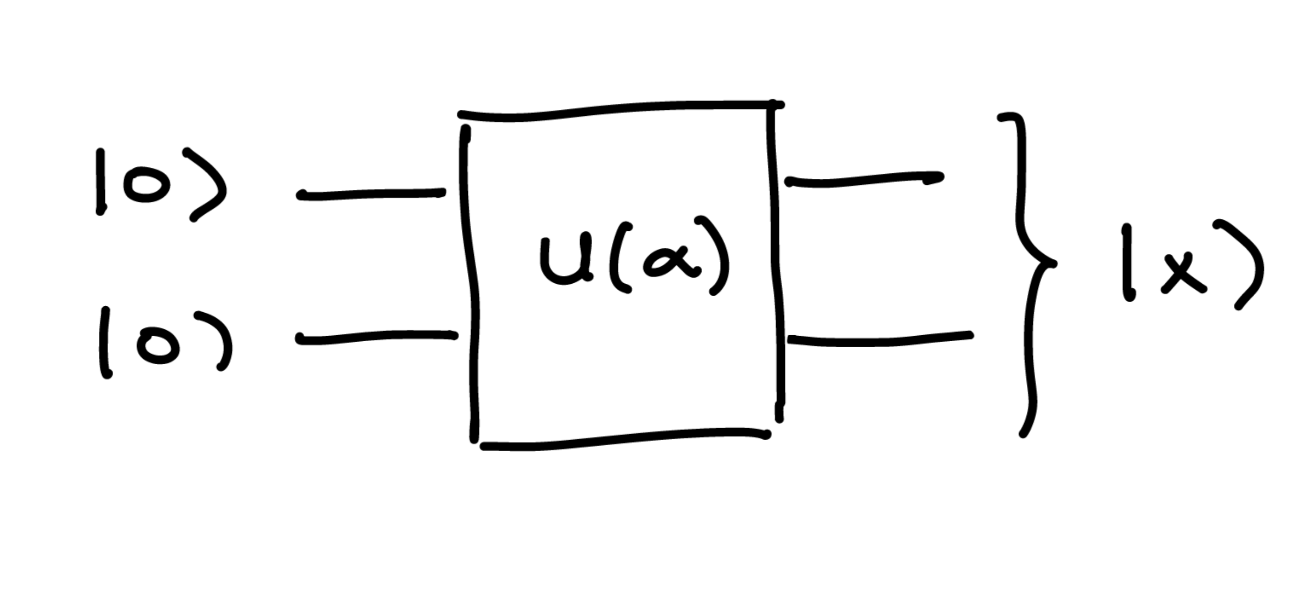
One of the computational basis states of 8 qubits

Data encoding

3	1
0	3

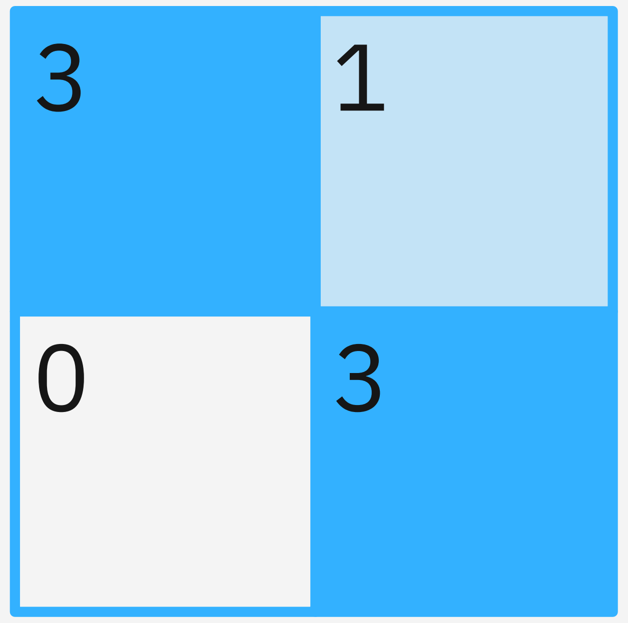
Amplitude encoding: Encode into quantum state amplitudes $|\psi_x\rangle = \sum_{i=1}^{N} x_i |i\rangle$

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3/\sqrt{19} \\ 1/\sqrt{19} \\ 0/\sqrt{19} \\ 3/\sqrt{19} \end{bmatrix}$$



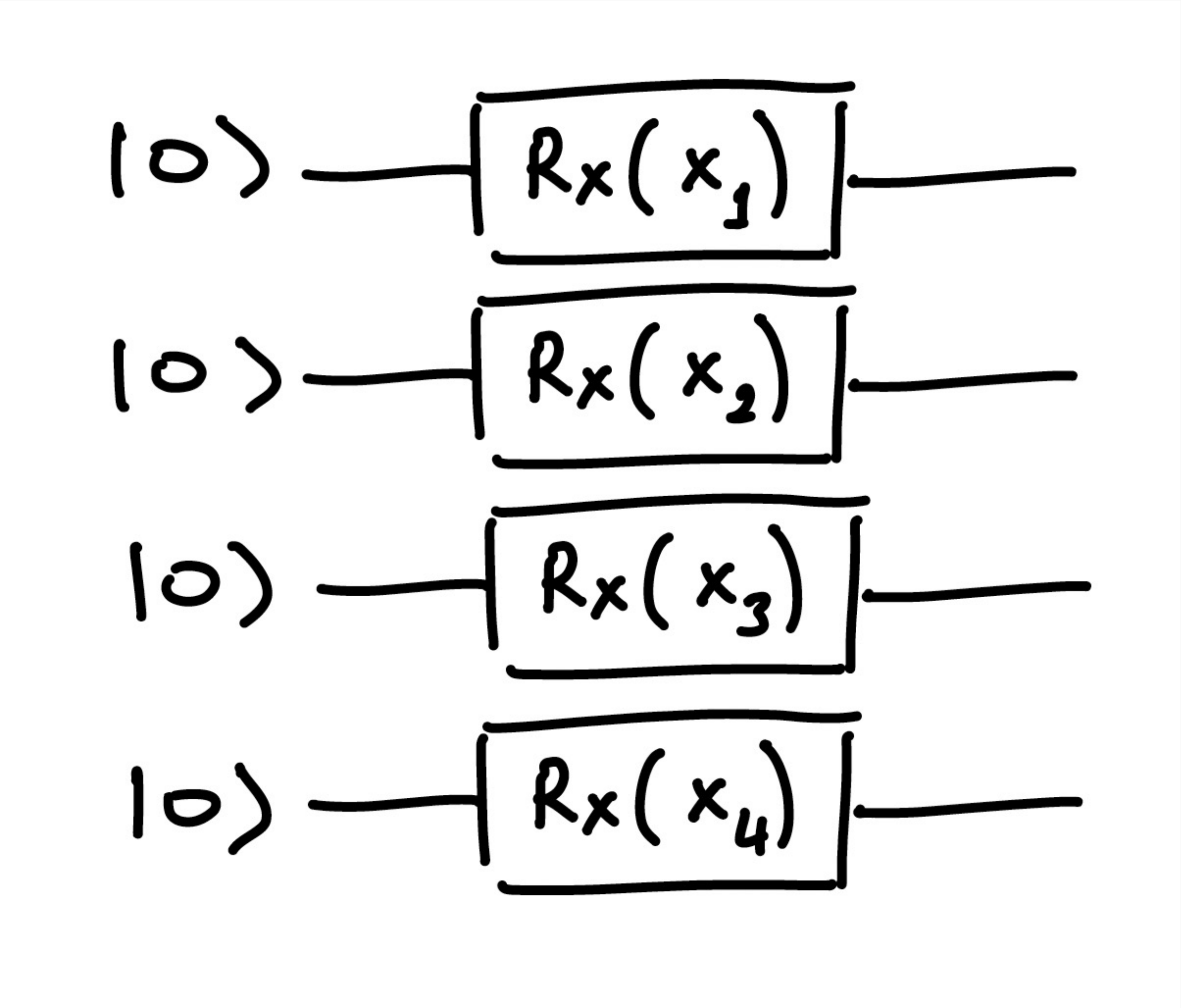
Amplitudes of 2 qubits

Data encoding



Angle encoding: Encode values into qubit rotation angles

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3/\sqrt{19} \\ 1/\sqrt{19} \\ 0/\sqrt{19} \\ 3/\sqrt{19} \end{bmatrix}$$



$$|x\rangle = \bigotimes_{i=1}^{N} \cos(x_i)|0\rangle + \sin(x_i)|1\rangle$$

angle encoding

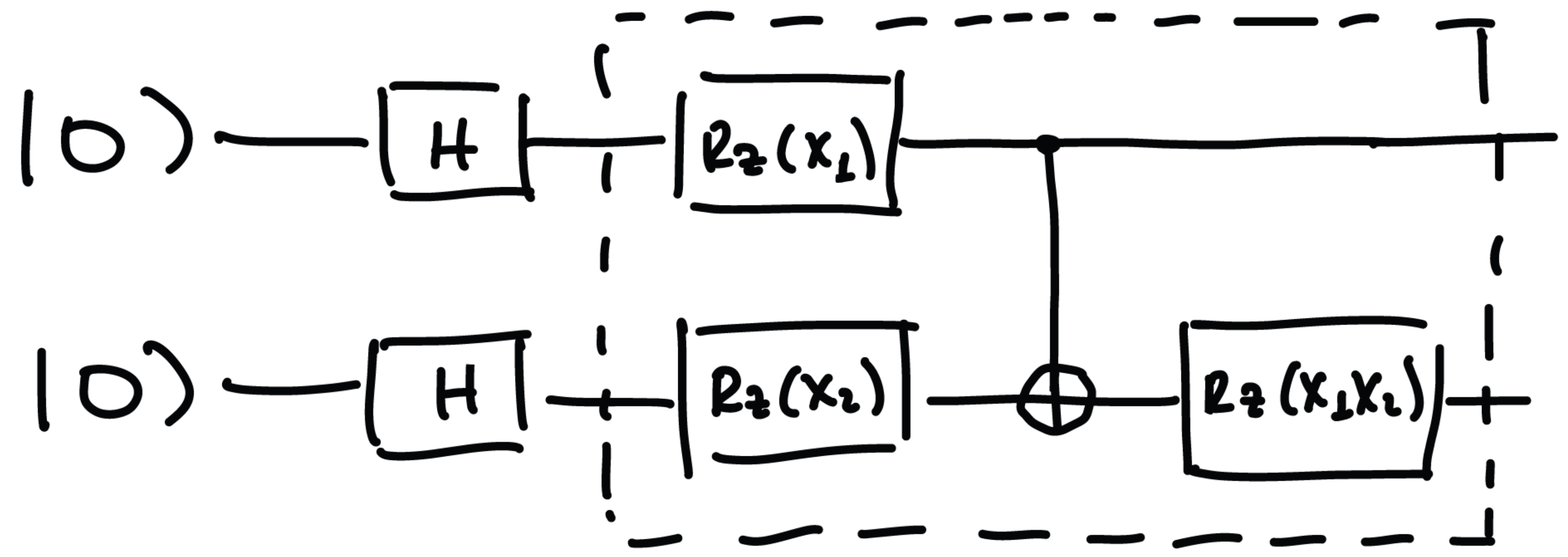
$$|x\rangle = \bigotimes_{i=1}^{n} \cos(x_{2i-1})|0\rangle + e^{ix_{2i}} \sin(x_{2i-1})|1\rangle$$

dense angle encoding

Data encoding

Higher order encoding: Feature maps

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$



blocks can be repeated

Havlíček, Vojtěch, et al. "Supervised learning with quantum-enhanced feature spaces." Nature 567.7747 (2019): 209-212.

Data encoding

Basis Encoding

Encode each n-bit feature into n qubits

$$x = (b_{n-1}, \dots, b_1, b_0) \quad \rightarrow \quad |x\rangle = |b_{n-1}, \dots, b, b_0\rangle$$

Amplitude Encoding

Encode into quantum state amplitudes

$$x = \begin{bmatrix} x_0 \\ \vdots \\ x_{N-1} \end{bmatrix} \quad \rightarrow \quad |\psi_x\rangle = \sum_{j=0}^{N-1} x_j |j\rangle$$

Angle Encoding

Encode values into qubit rotation angles

$$|x\rangle = \bigotimes^{N} \cos(x_i)\,|0\rangle + \sin(x_i)\,|1\rangle$$

Arbitrary Encoding (Feature Map)

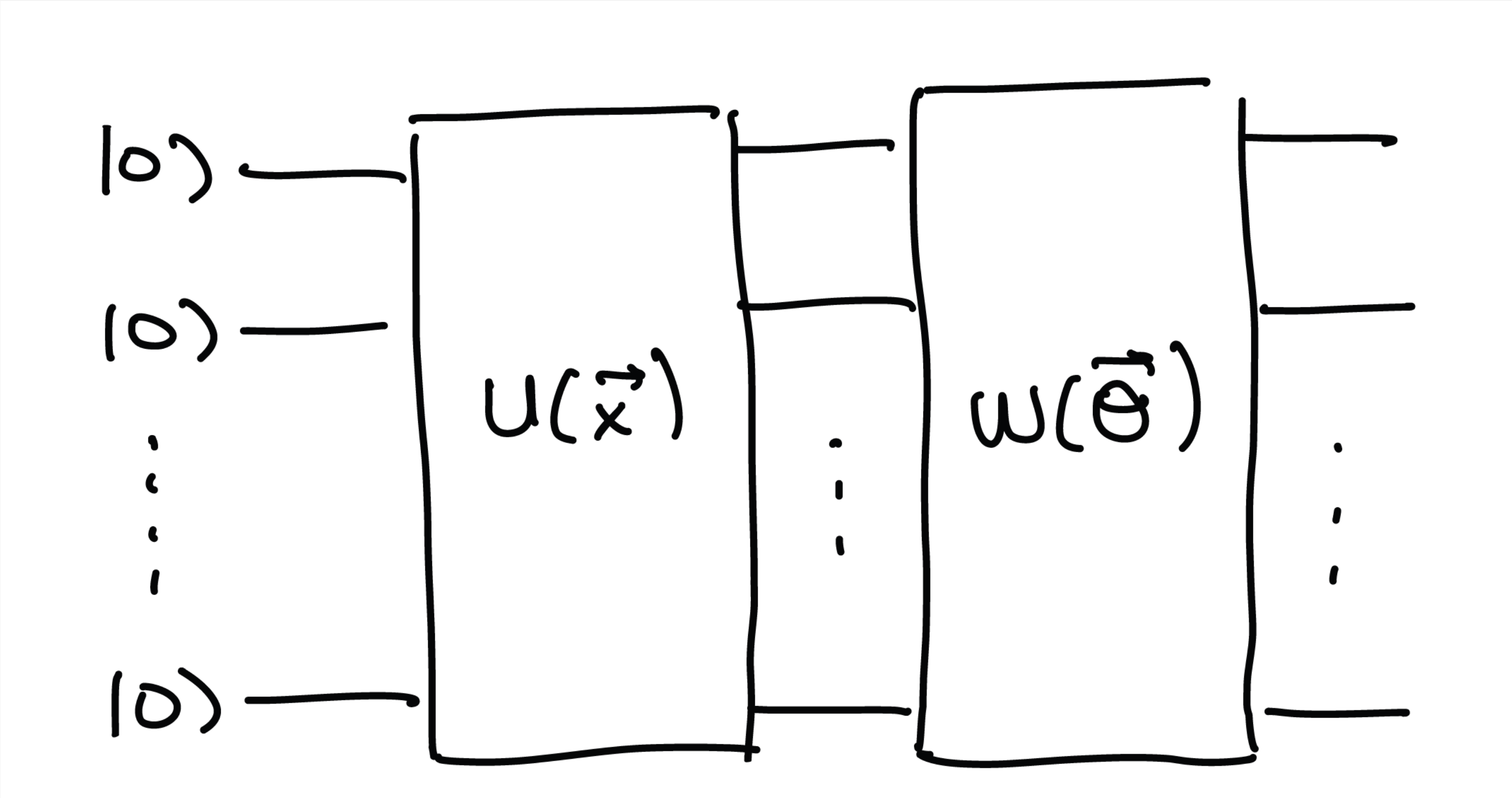
Encode N features on N rotation gates in constant-depth circuit with n qubits

$$x = \begin{bmatrix} x_0 \\ \vdots \\ x_{N-1} \end{bmatrix} \quad \rightarrow \quad |\psi_x\rangle = \mathcal{U}_{\Phi(x)}|0\rangle$$

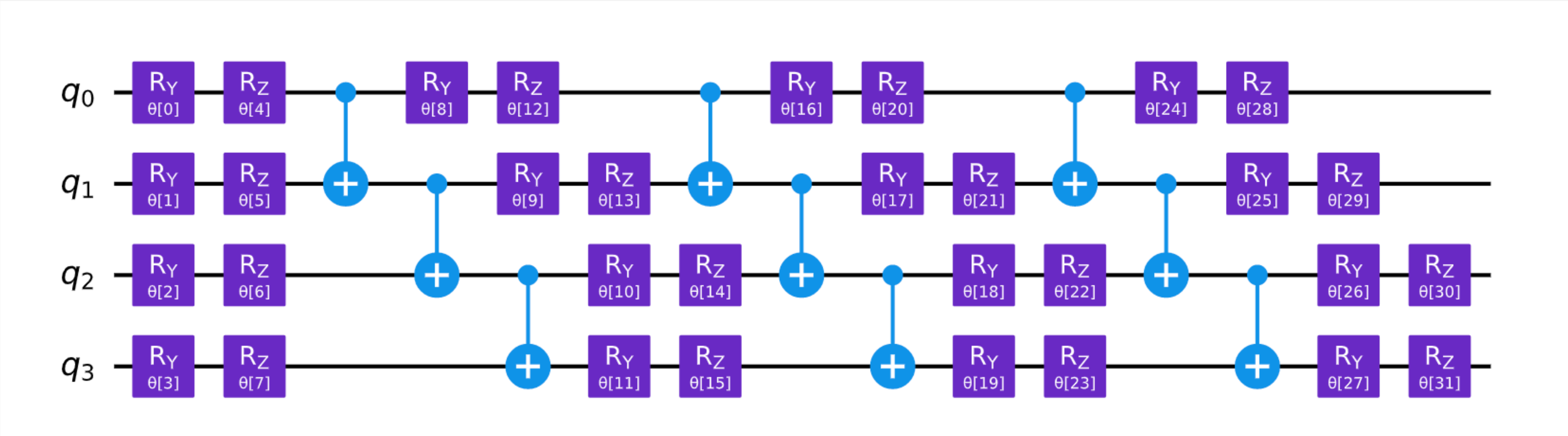
Encoding	# Qubits	State prep runtime
Basis	nN	$O(N)$
Amplitude	$\log(N)$	$\dfrac{O(N)}{O(\log(N))}$
Angle	N	$O(N)$
Arbitrary	n	$O(N)$

N features each

Variational model



Data encoding Ansatz

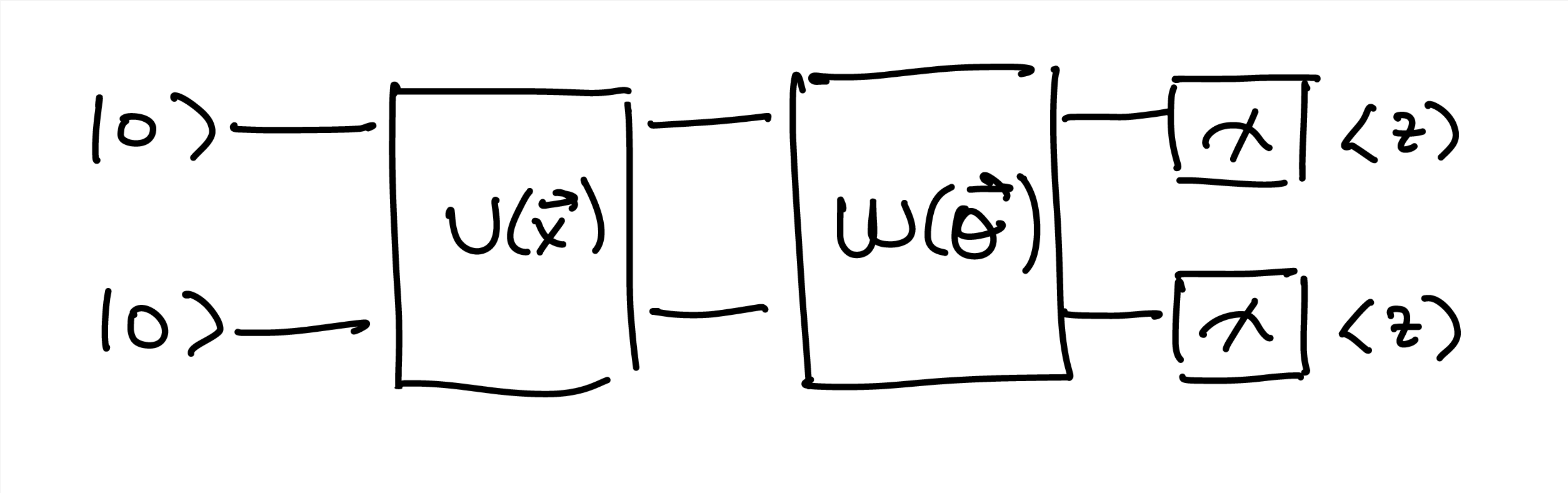


Goal: designing a **hardware-efficient** ansatz

expressivity and depth

Leone, Lorenzo, et al. "On the practical usefulness of the hardware efficient ansatz." arXiv preprint arXiv:2211.01477 (2022).

Extracting labels



measurement outcomes → labels → cost function → classical optimizer

Binary classification {1,-1}:	Qiskit
1. Parity post-processing (00, 01, 10, 11)	*sampler*
2. Measure only 1 qubit (<Z> >= 0, otherwise)	*estimator*

Optimization: parameter update

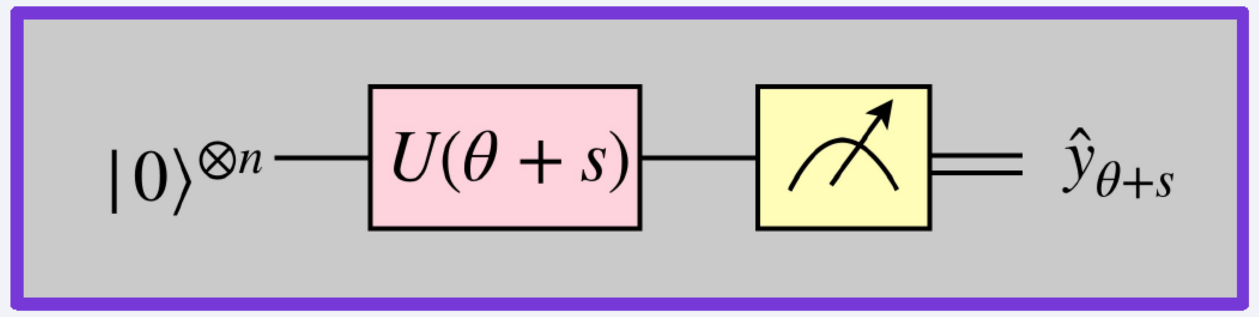
e.g. Mean squared error

Cost: $$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2.$$

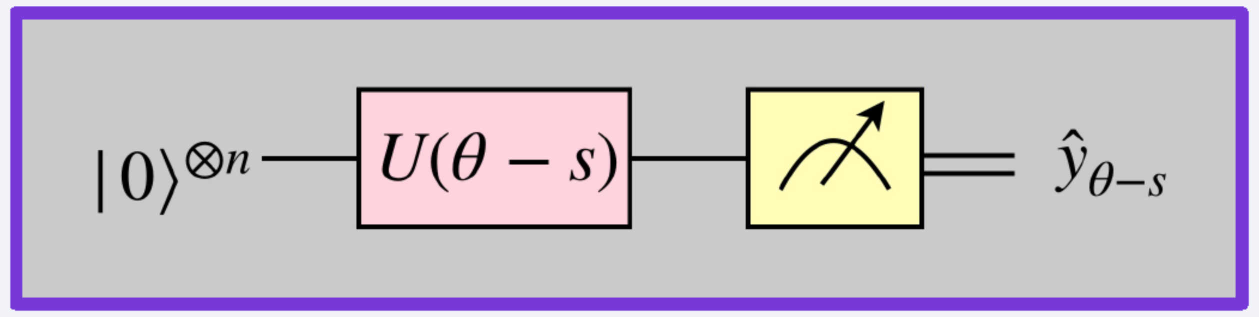
If optimizer needs: $\partial_{\theta_i} f(\boldsymbol{\theta})$

Parameter-shift rule

Gradient =



−



/ 2

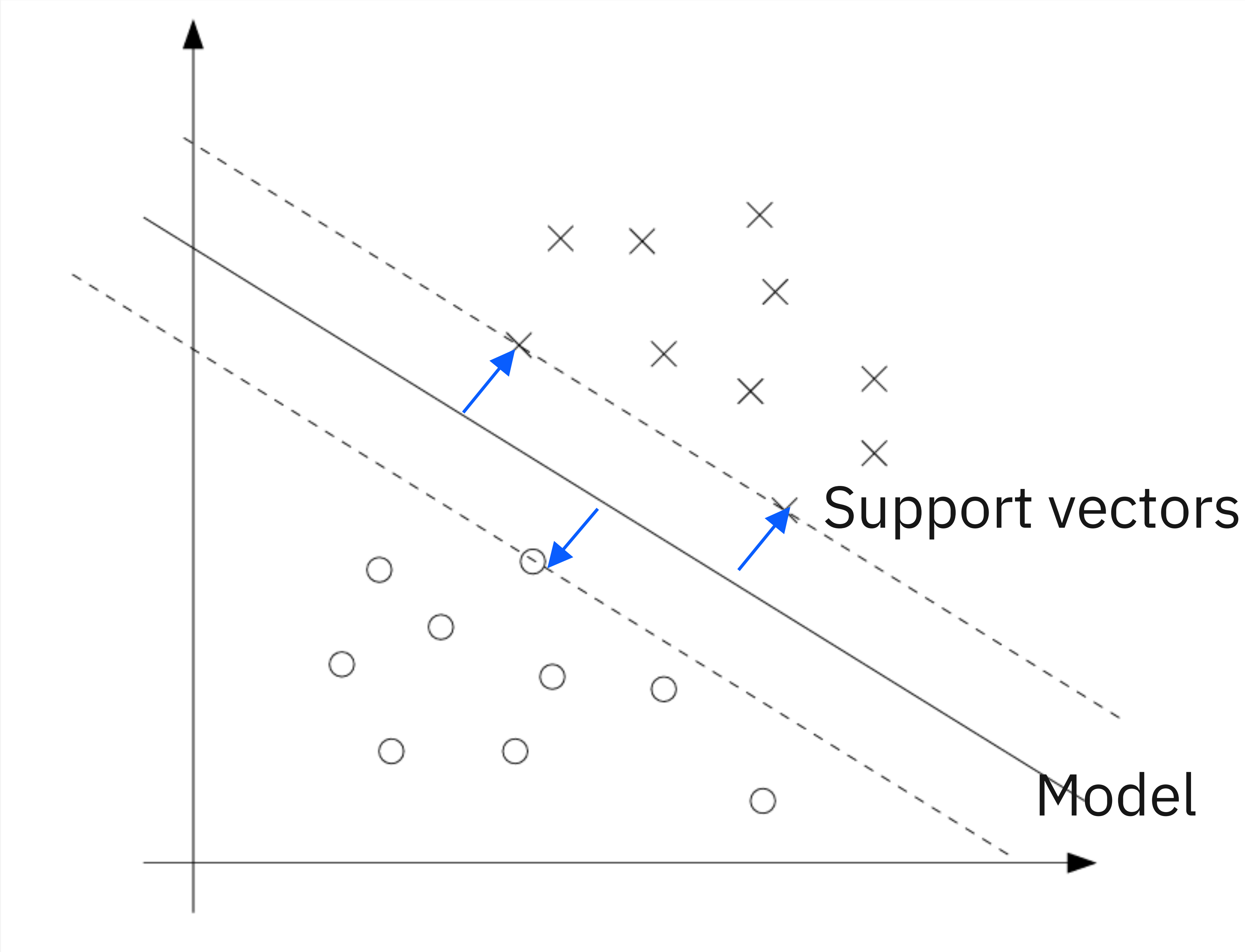
$s = \pi/2$

Remark: SPSA (Simultaneous Perturbation Stochastic Approximation)

Schuld, Maria, et al. "Circuit-centric quantum classifiers." Physical Review A 101.3 (2020): 032308.

Quantum kernels and support vector machines

Support vector machines (SVMs)



Classification problem, e.g. binary classification

– Primal formulation

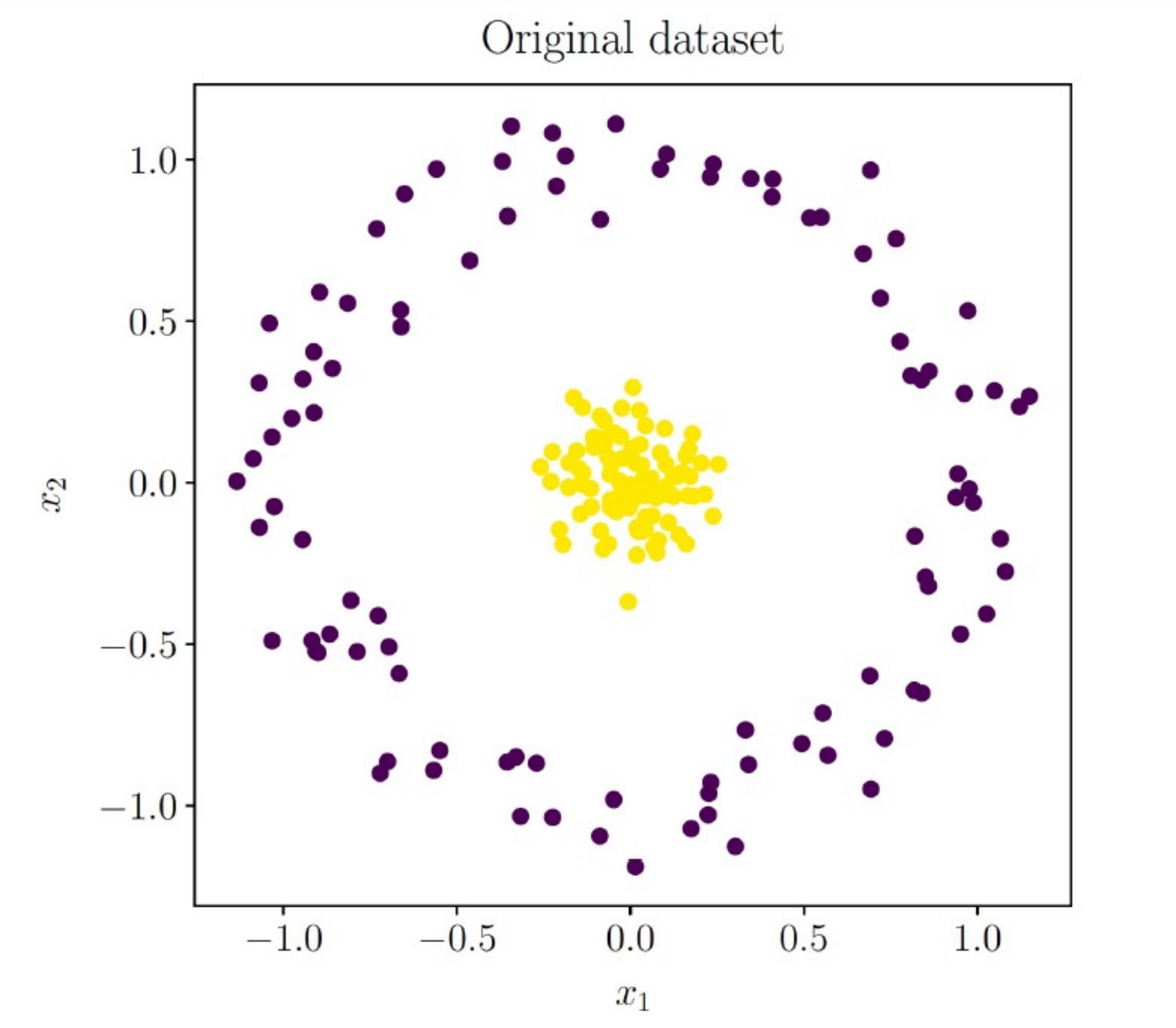
$$f(x) = \vec{\theta}^T \vec{x} + b$$

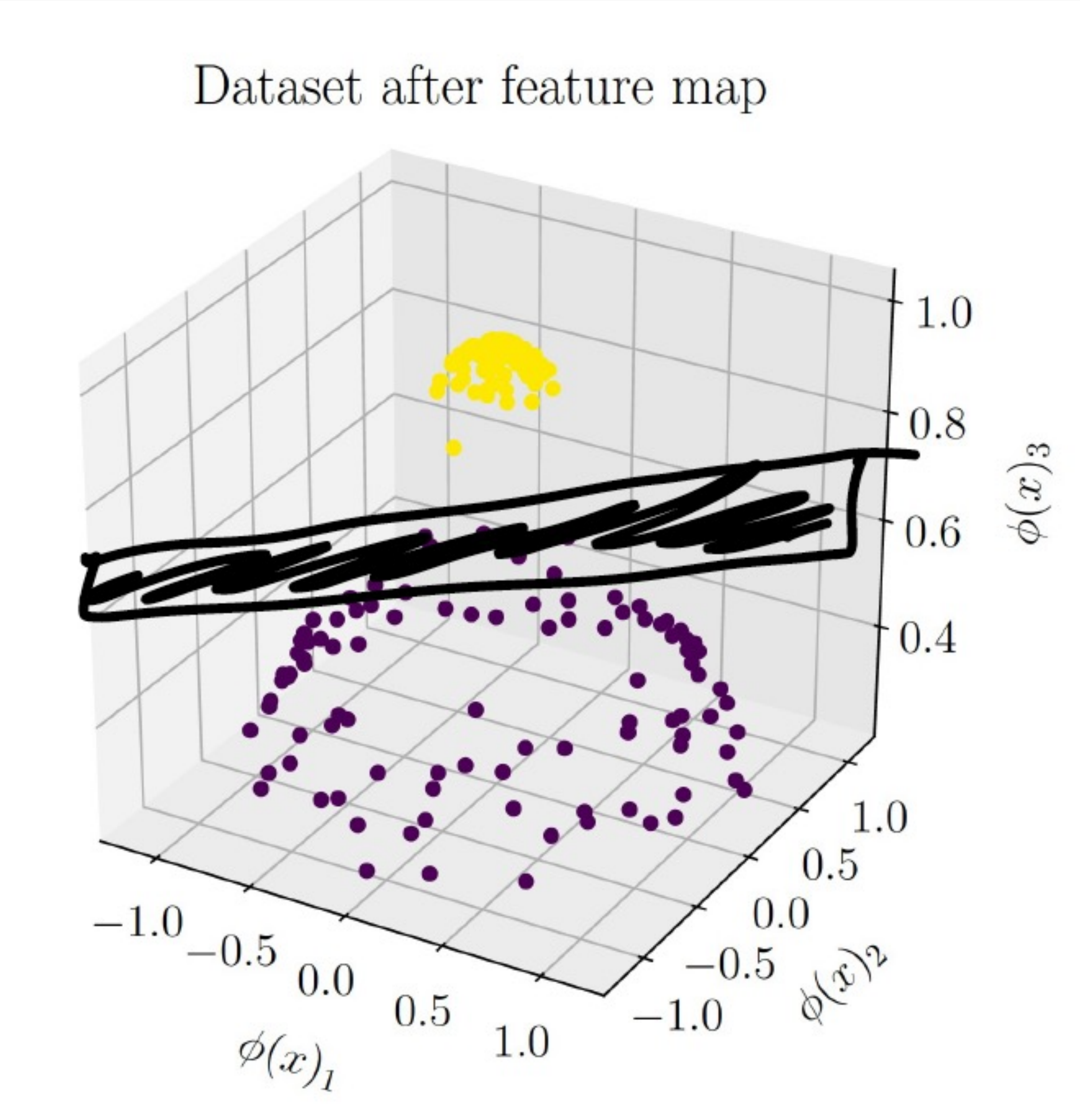
– Dual formulation

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i (\vec{x}_i^T \vec{x}) + b$$

Support vector machines

When data is not linearly separable





$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\phi(\vec{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1 x_2 \end{bmatrix}$$

feature map

Support vector machines

Primal formulation

$$f(x) = \vec{\theta}^T \vec{x} + b$$

$$f(x) = \theta^T \phi(x) + b$$

Dual formulation

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i \left(\vec{x}_i^T \vec{x} \right) + b$$

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i \left(\phi(x_i)^T \phi(x) \right) + b$$

inner product
"kernel"

Quantum kernels

- Interpreting data encoding to a quantum state as a feature map $x \rightarrow |\phi(x)\rangle$

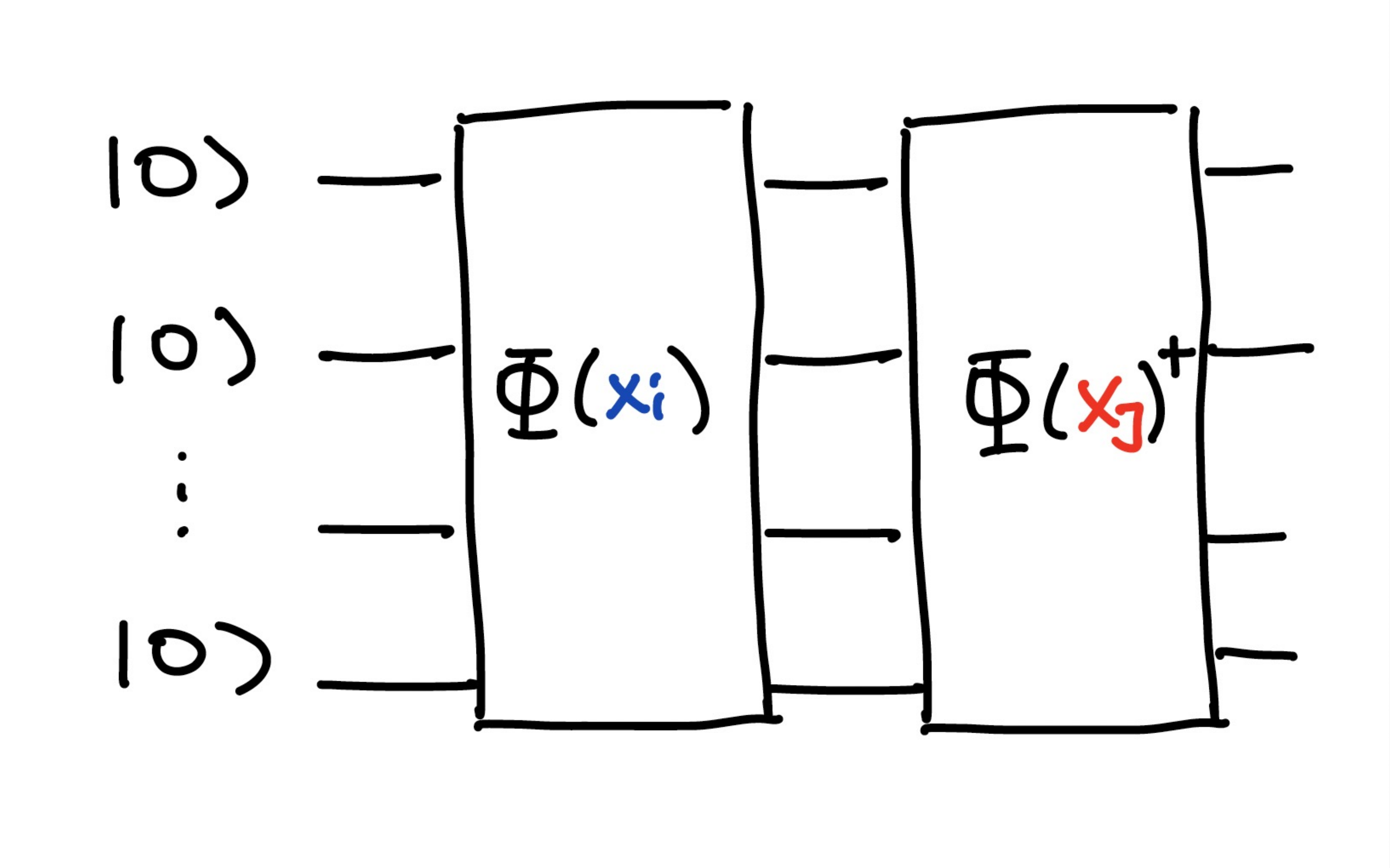
- Quantum kernels can only be expected to do better than classical kernels if they are **hard to estimate classically.**
  - necessary but not sufficient

- It was shown that learning problems **exist**, for which learners with access to quantum kernel methods have a quantum advantage over all classical learners.

Havlíček, Vojtěch, et al. "Supervised learning with quantum-enhanced feature spaces." Nature 567.7747 (2019): 209-212.

Glick, Jennifer R., et al. "Covariant quantum kernels for data with group structure." Nature Physics (2024): 1-5.

Quantum SVM



quantum kernel estimator

$$K_{i,j} = |\langle \Phi(x_j) | \Phi(x_i) \rangle|^2$$

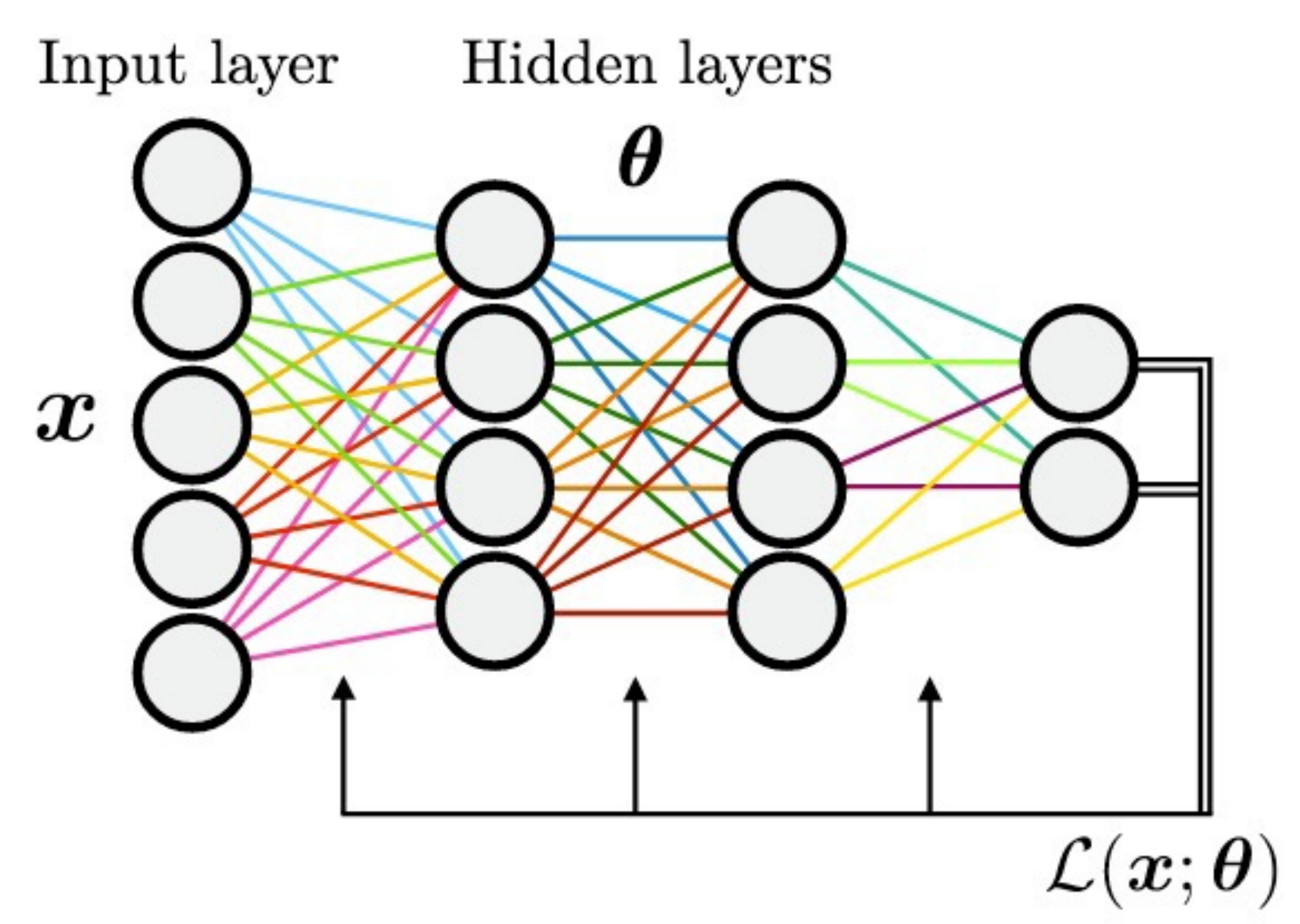
$$\Pr[\text{measure } |0\rangle] = |\langle 0| \Phi(x_j)^\dagger \Phi(x_i) |0\rangle)|^2$$

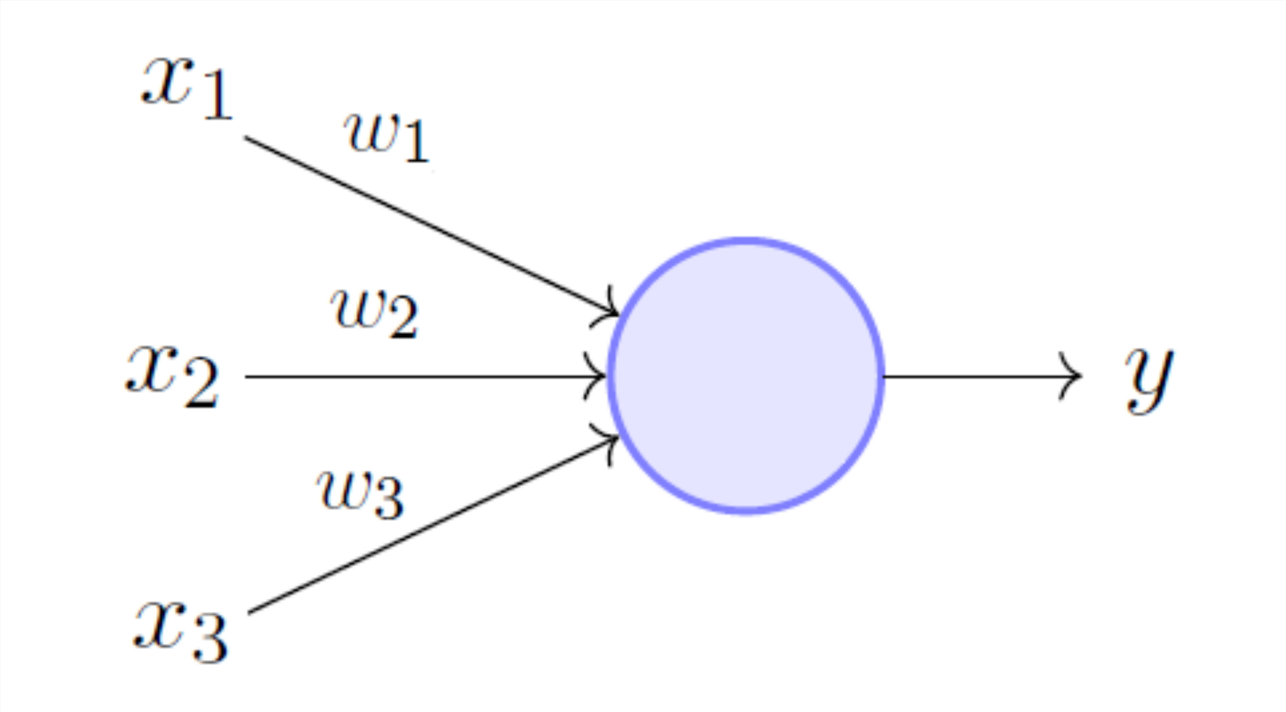
For i,j in the training set:

- Prepare $\Phi(x_j)^\dagger \Phi(x_i) |0\rangle$
- Let $K_{i,j} = \Pr[\text{measure } |0\rangle]$
- Plug $K_{i,j}$ into the dual form and solve
- Return $\{\alpha_i\}$
- Label $\text{label}(s) = \text{sign}\left(\sum_i \alpha_i K(x_i, s) + b\right)$

Quantum neural networks

Classical feed-forward neural networks





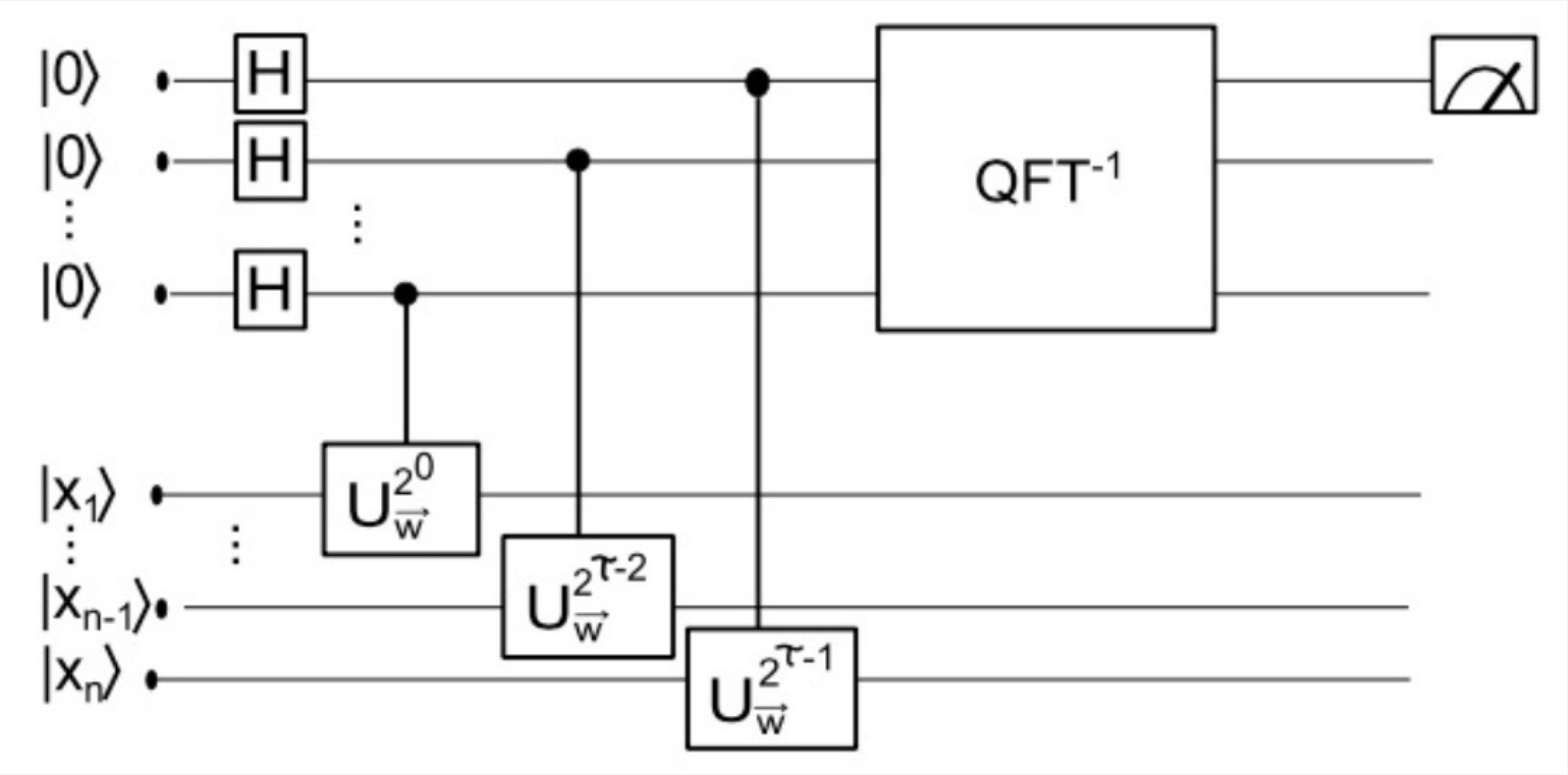
perceptron

$$f(\vec{x}) = \sigma(\vec{w} \cdot \vec{x} + b)$$

non-linear activation function

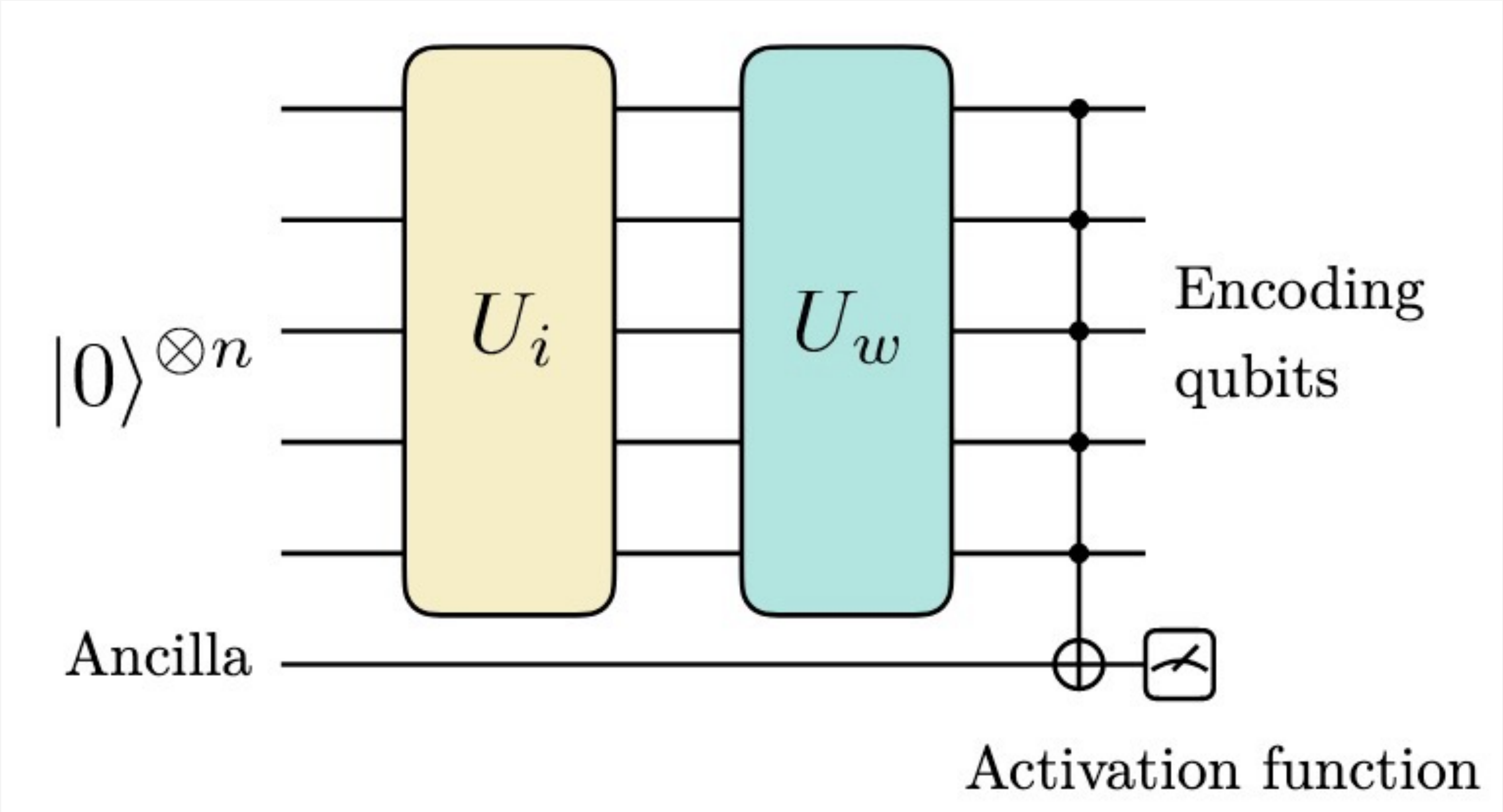
Quantum perceptron

Need to implement non-linearity with quantum circuits



QFT based perceptron

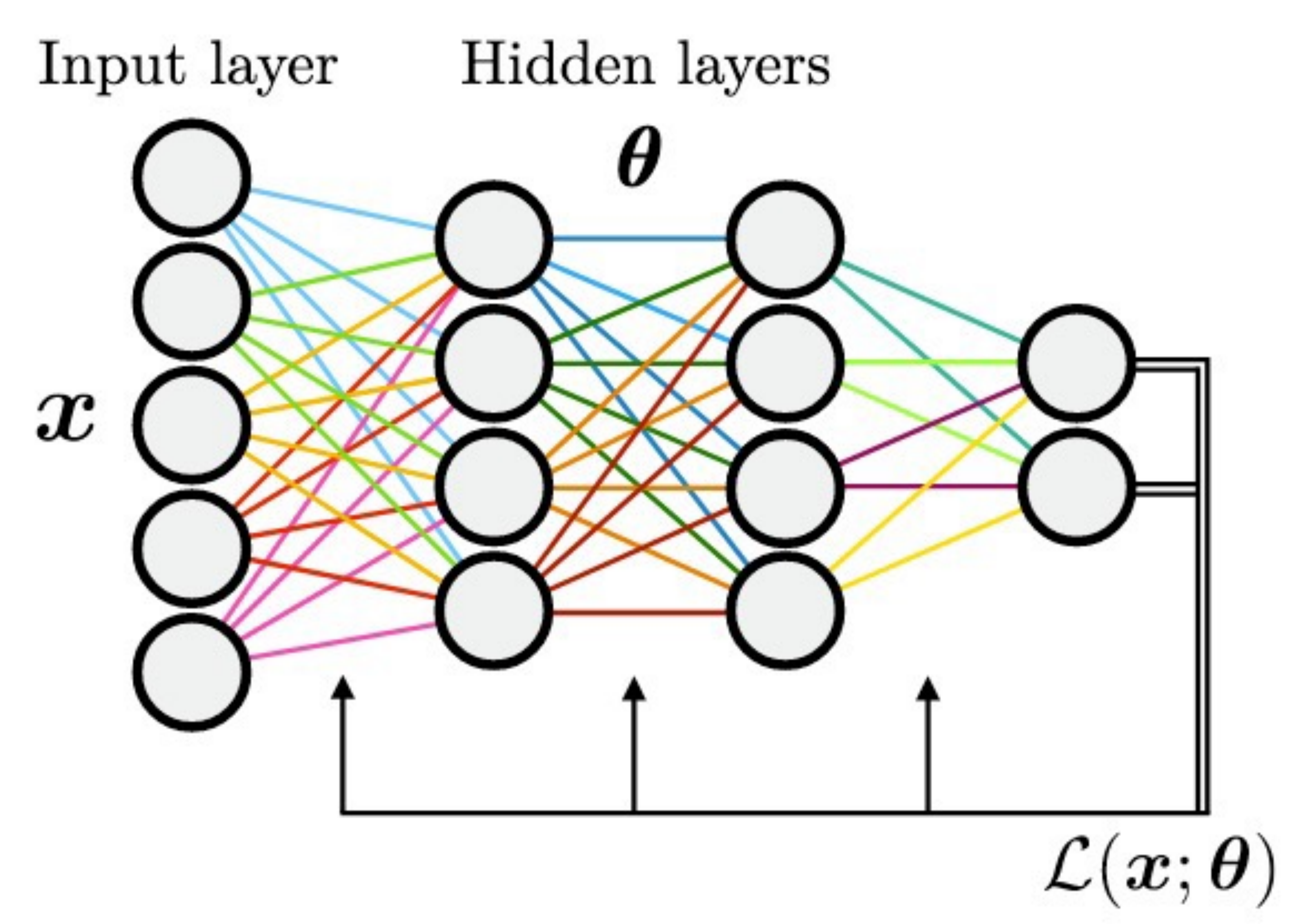
M. Schuld et al., Phys. Lett. A 379, 660 (2015)



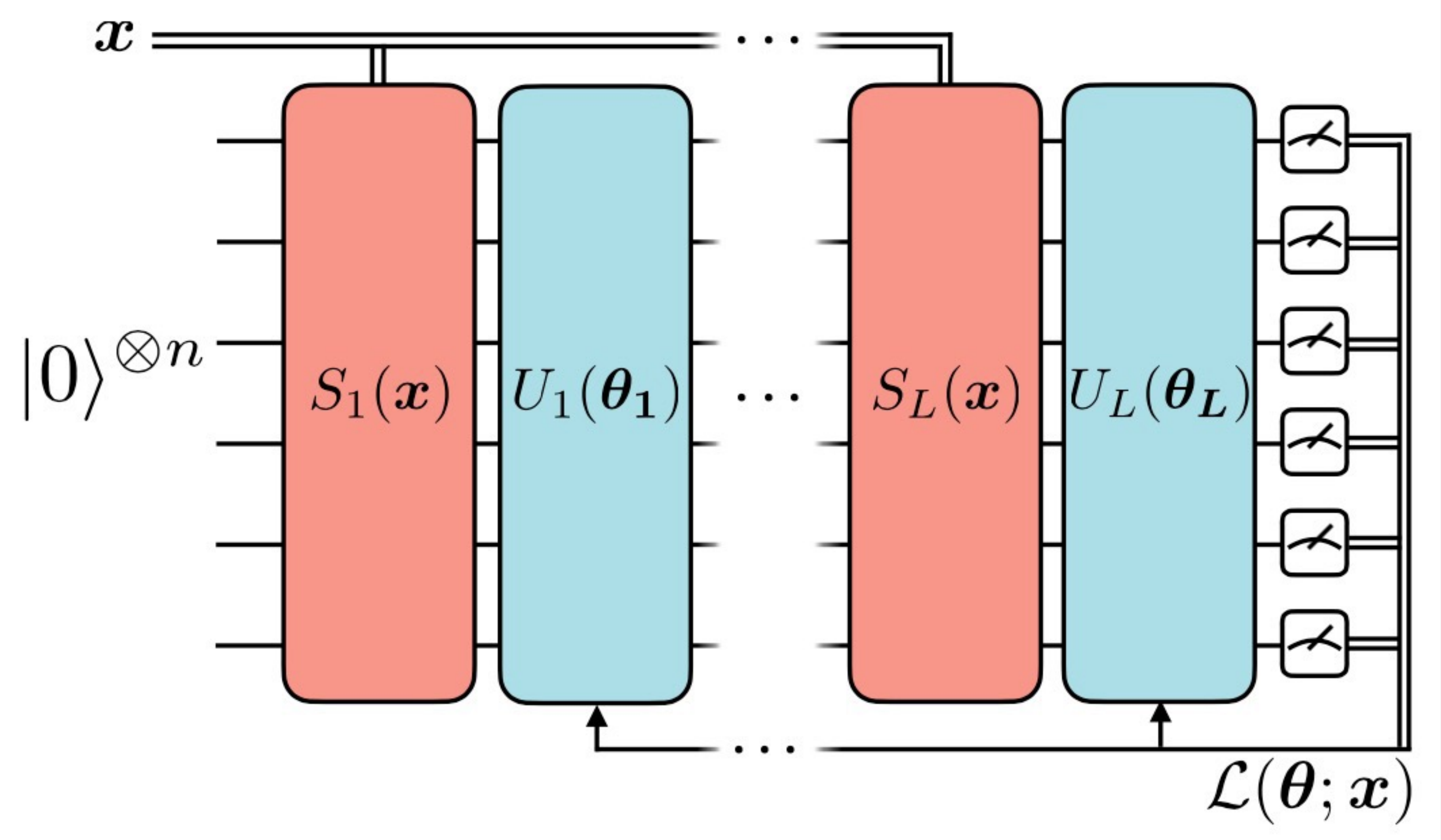
Non-linearity from measurement

F. Tacchino et al., npj Quantum Inf. 5, 26 (2019)

Quantum neural networks



classical feed-forward neural network



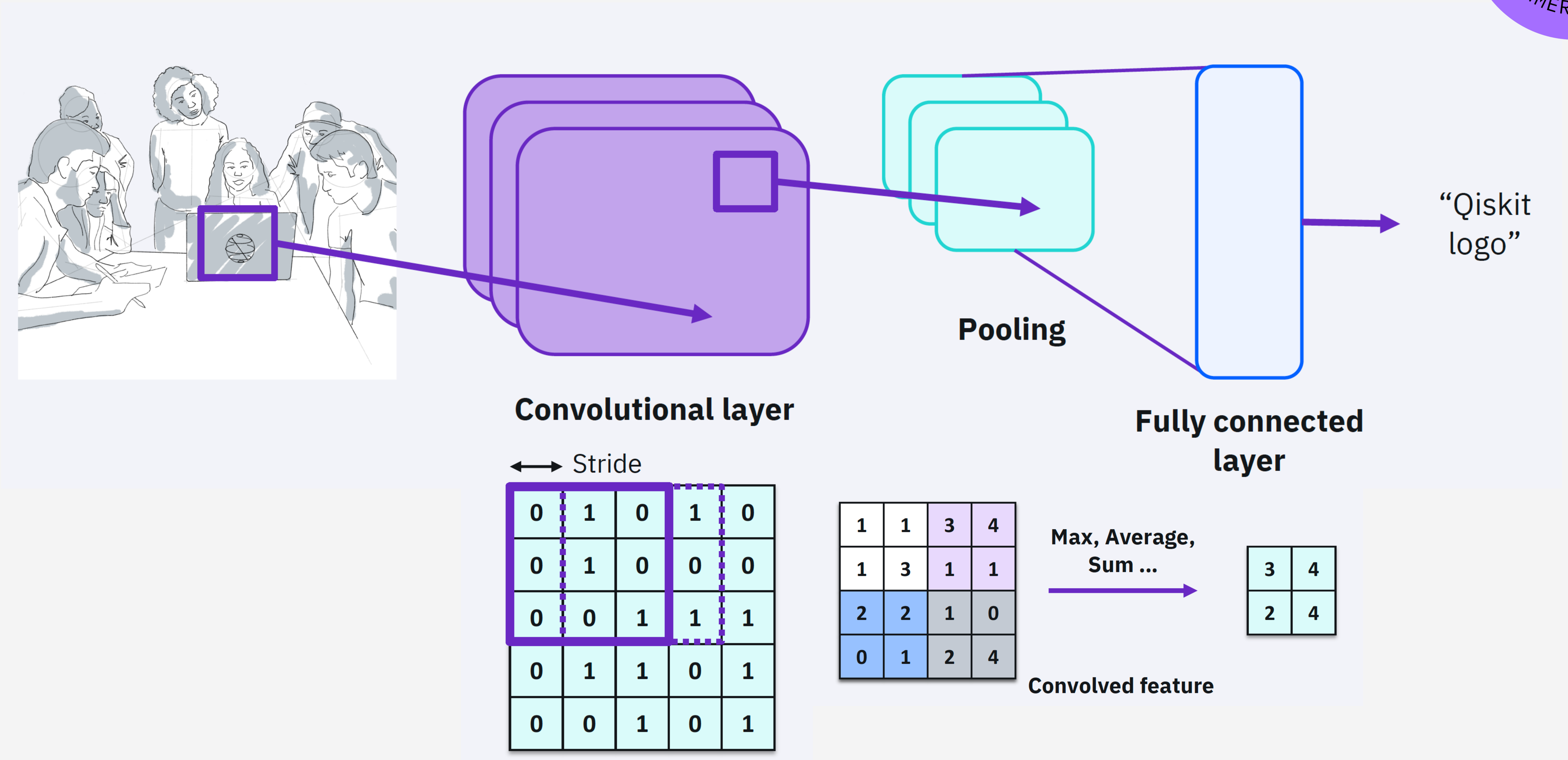
quantum neural networks

difference: data reuploading

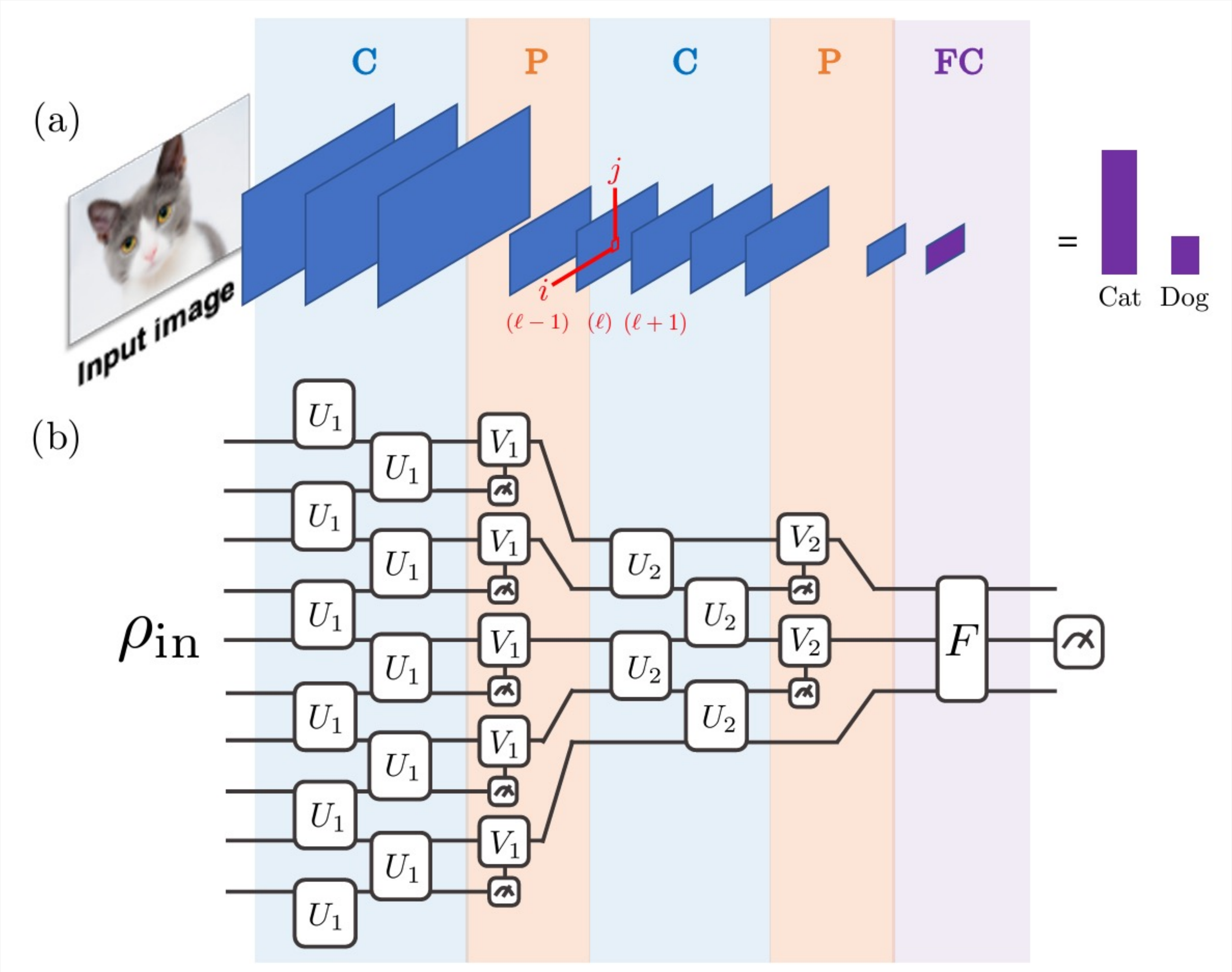
universal function approximators

Pérez-Salinas, Adrián, et al. "Data re-uploading for a universal quantum classifier." Quantum 4 (2020): 226.

Convolutional neural networks (CNNs)



Quantum convolutional neural networks (QCNNs)



Properties:

- QCNNs have *O(logN)* layers and parameters
- They don't suffer from the problem of barren plateaus

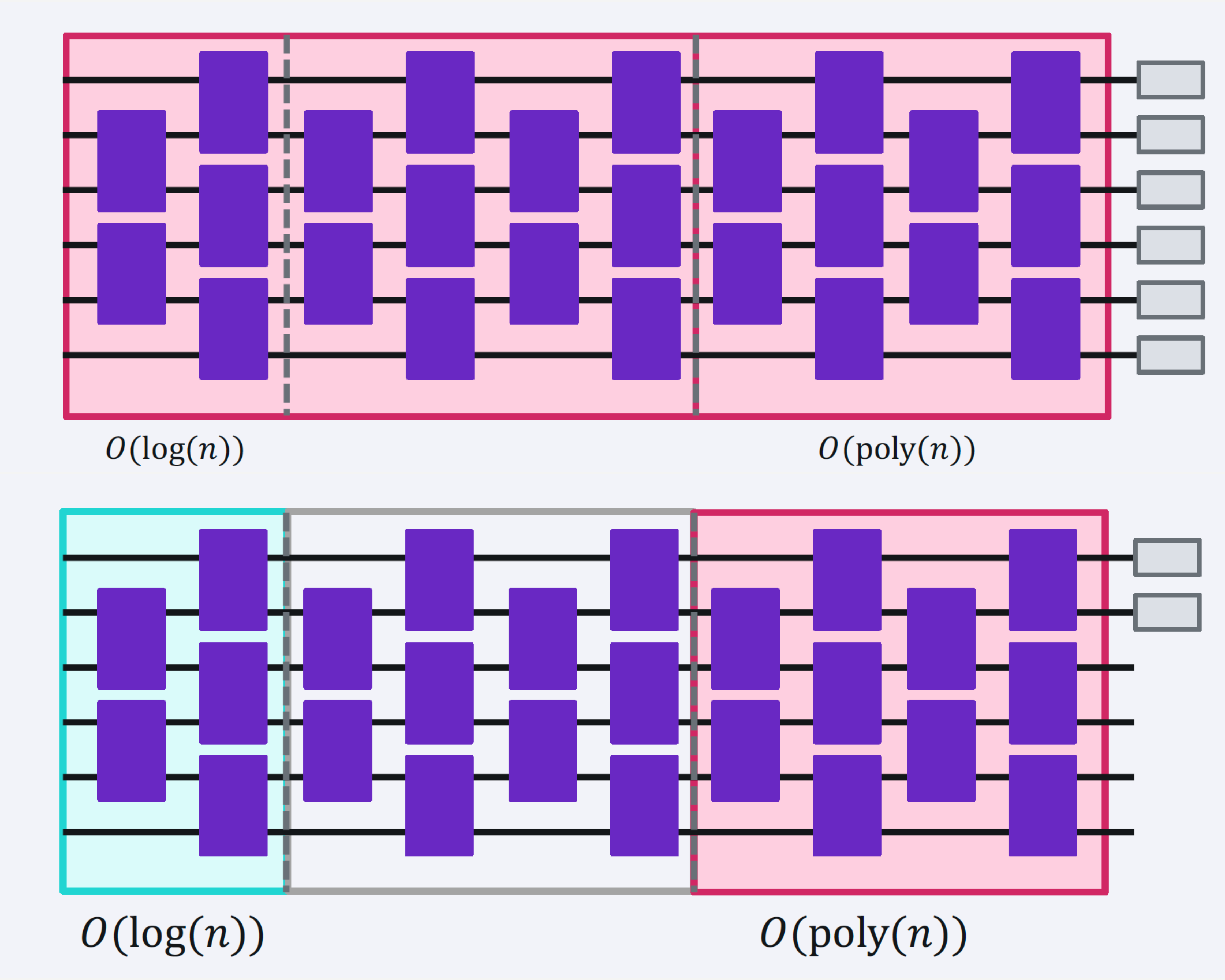
Pesah, Arthur, et al. "Absence of barren plateaus in quantum convolutional neural networks." Physical Review X 11.4 (2021): 041011.

Cong, Iris, Soonwon Choi, and Mikhail D. Lukin. "Quantum convolutional neural networks." Nature Physics 15.12 (2019): 1273-1278.

Barren plateaus

Random initialization of parameters in deep circuits:

Gradients of the cost function **vanish** exponentially with the number of qubits



Mitigating barren plateaus:

Initialization strategies

For an alternating layered ansatz:

shallow circuits

local cost functions

McClean, Jarrod R., et al. "Barren plateaus in quantum neural network training landscapes." Nature communications 9.1 (2018): 4812
Cerezo, Marco, et al. "Cost function dependent barren plateaus in shallow parametrized quantum circuits." Nature communications 12.1 (2021): 1791..

Tutorial session

Now get your notebook and open it.

You can open it either on your local or on qBraid Lab.

Table of Contents:

NTU Road to Quantum Utility Workshop 2025

Quantum kernel estimation

Boseong Kim (January 21, 2025)

This code is based on "Quantum kernel estimation of Quantum Machine Learning" of IBM Quantum Learning which will be released soon.

Single kernel matrix entry

Step 1: Map classical inputs to a quantum problem

Let us first consider a data set with just a few features, say 10. We need at least two points to calculate the kernel matrix elements, so we will start with that, and import a few needed packages:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

#Two mock data points, including category labels, as in training
small_data = [[-0.194,0.114,-0.006,0.301,-0.359,-0.088,-0.156,0.342,-0.016,0.143,1],[-0.1,0.0

#Data points with labels removed, for inner product
train_data = [small_data[0][:-1],small_data[1][:-1]]
```

IBM