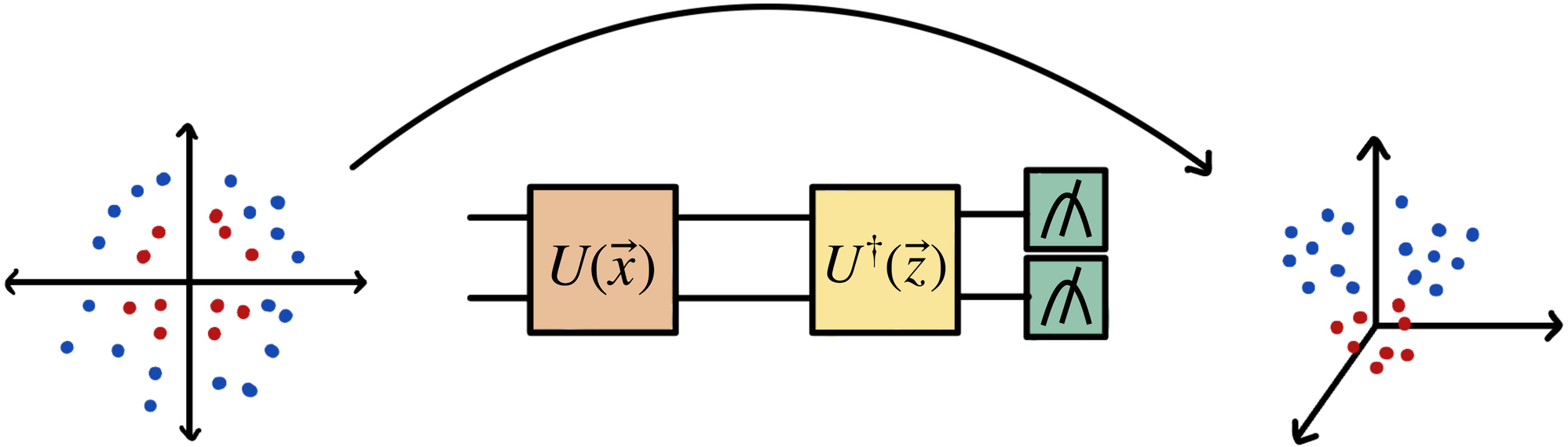
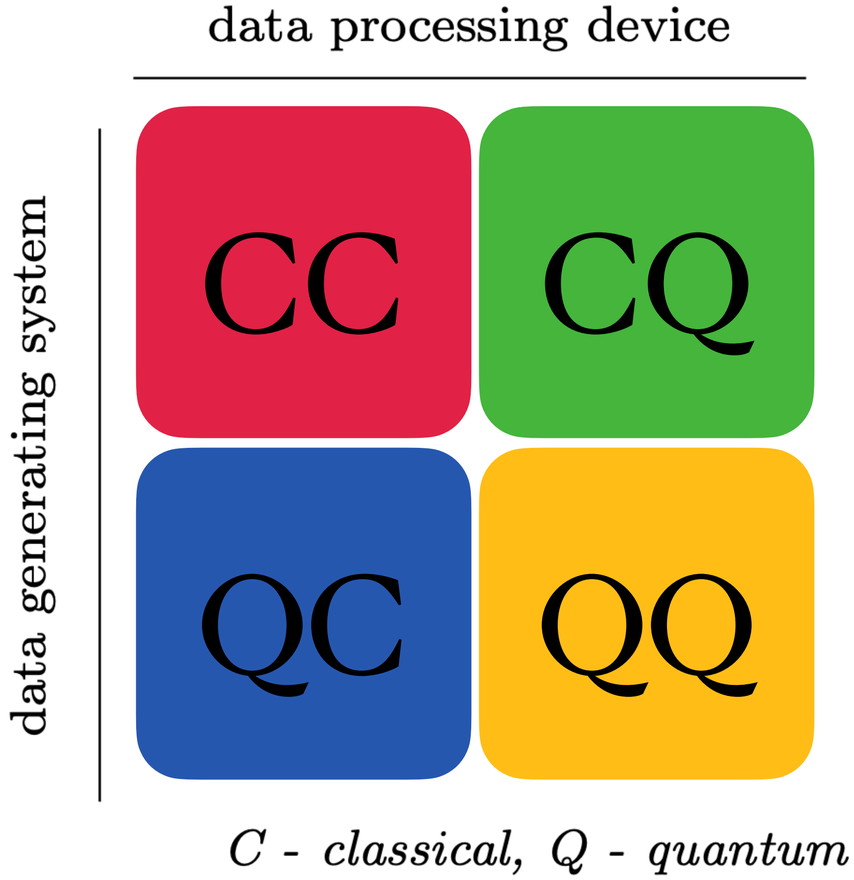
QUANTUM KERNEL METHODS



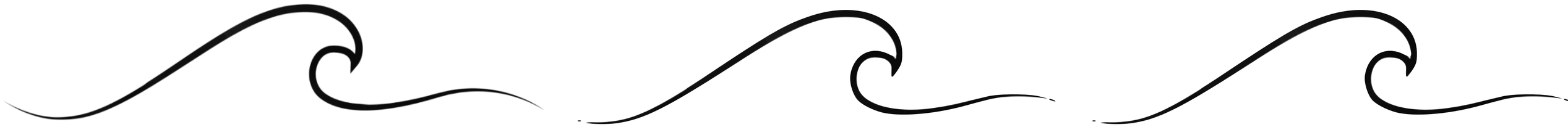
Spring School on Theoretical and Computational Foundations of Quantum Technologies

WHAT IS QUANTUM MACHINE LEARNING?

Quantum Machine Learning seeks to combine techniques or theory from quantum mechanics and machine learning.



THREE 'WAVES' IN QUANTUM MACHINE LEARNING



1

Developing QML Algorithms
for
Fault-Tolerant Settings

2

Developing QML Algorithms
for the
NISQ Era

3

Theoretical Studies
of
QML Algorithms

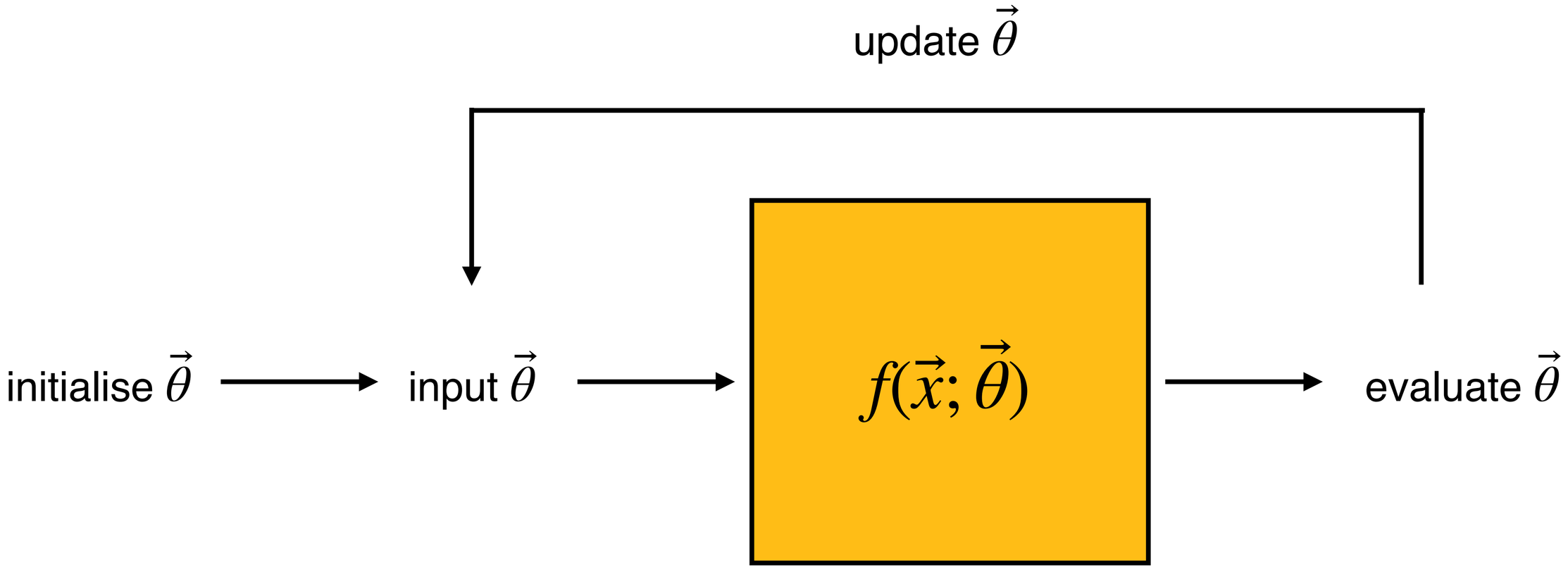
TWO POPULAR APPROACHES IN QML

Quantum ‘Neural Networks’

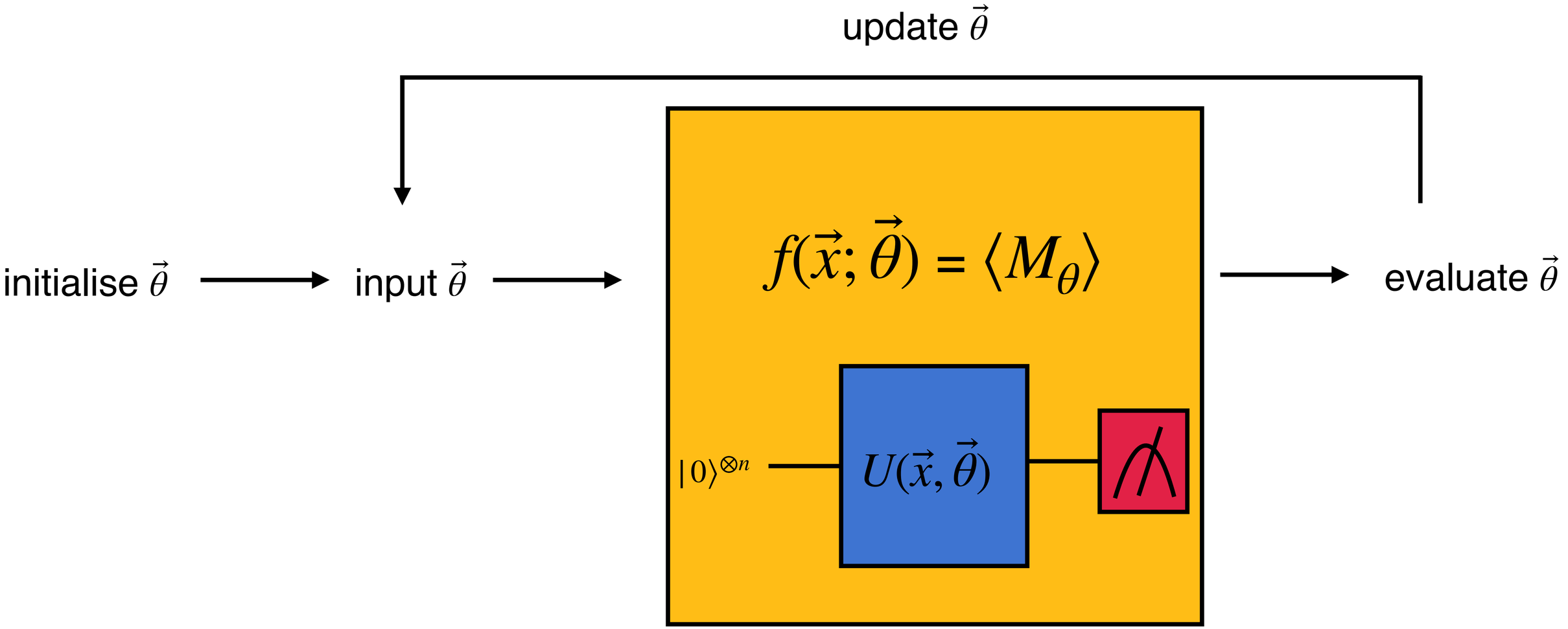
and

Quantum Kernel Methods

CLASSICAL NEURAL NETWORKS

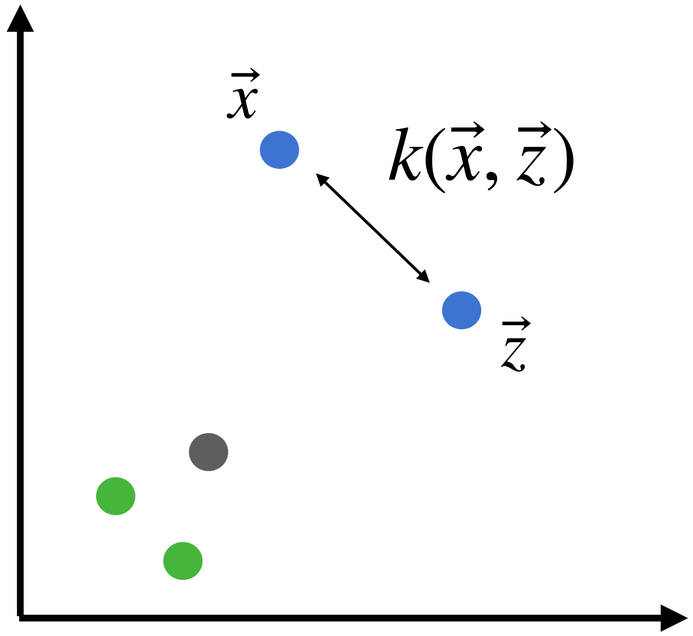


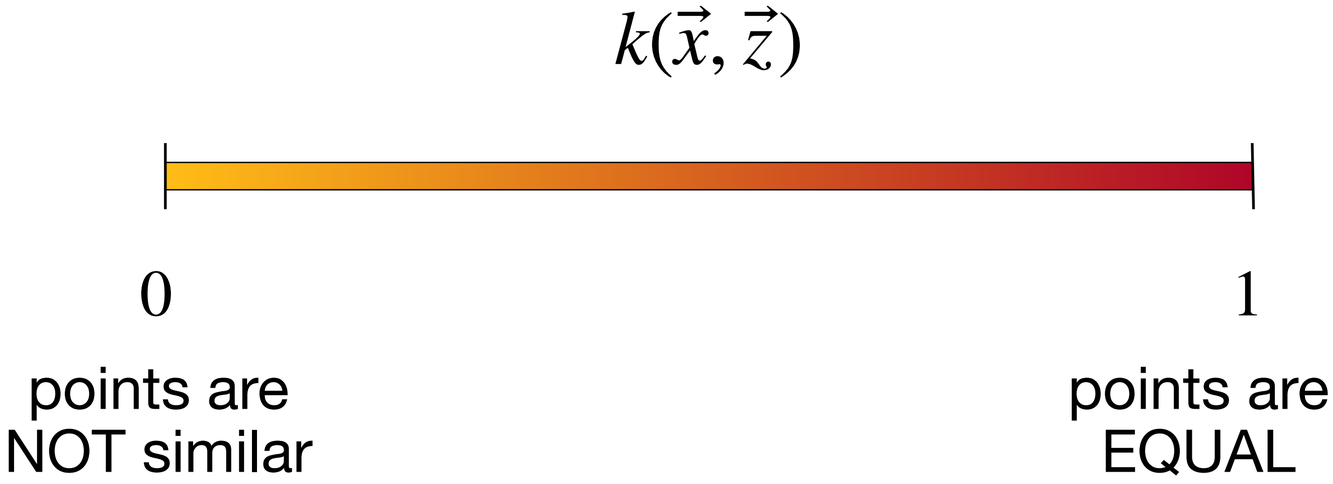
QUANTUM 'NEURAL NETWORKS'



CLASSICAL KERNEL METHODS

Kernel methods solve machine learning tasks based on the idea of a **similarity measure** between data points. The similarity measure is referred to as a **kernel**.





CLASSICAL KERNEL METHODS

A kernel is some similarity measure defined over the input space.

Given an input space X, a kernel is a positive semi-definite bivariate function

$$k : X \times X \to \mathbb{R}$$

The kernel can be used to construct a Kernel Matrix (Gram matrix). Given a dataset $D = \{\overrightarrow{x_1}, \ldots, \overrightarrow{x_2}\}$, the Kernel Matrix is

$$K_{i,j} = k(\overrightarrow{x_i}, \overrightarrow{x_j})$$

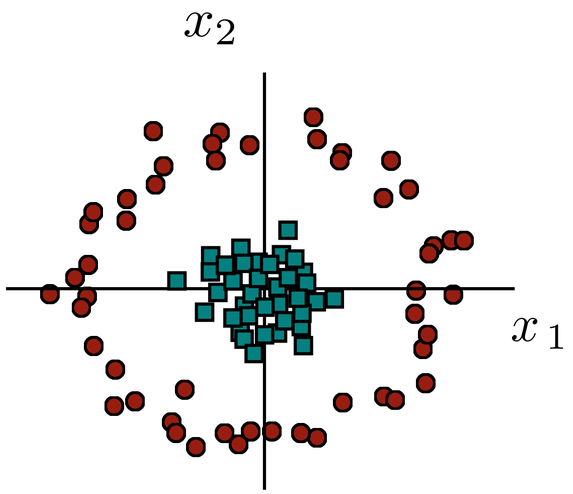
The Kernel Matrix is symmetric, meaning

$$K_{i,j} = K_{j,i}$$

KERNELS ARE CLOSELY LINKED TO FEATURE MAPS

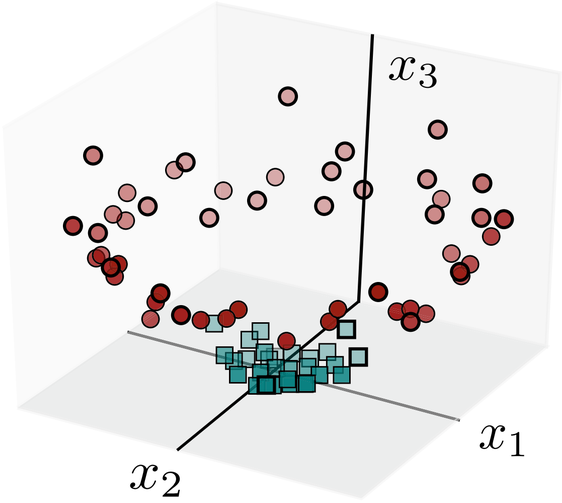
For every kernel function, we are guaranteed that there exists at least one feature map $\phi : X \to F$ that maps the input data into a suitable feature space F such that

$$k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \,|\, \phi(\mathbf{z}) \rangle$$



X

ϕ



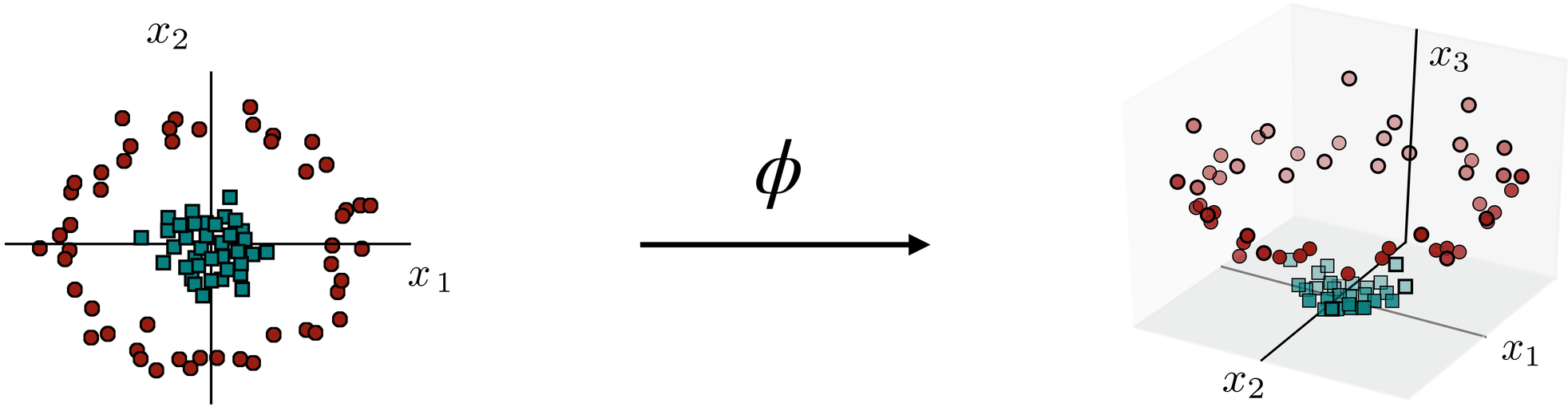
F

How could we define ϕ?

KERNELS ARE CLOSELY LINKED TO FEATURE MAPS

For every kernel function, we are guaranteed that there exists at least one feature map $\phi : X \to F$ that maps the input data into a suitable feature space F such that

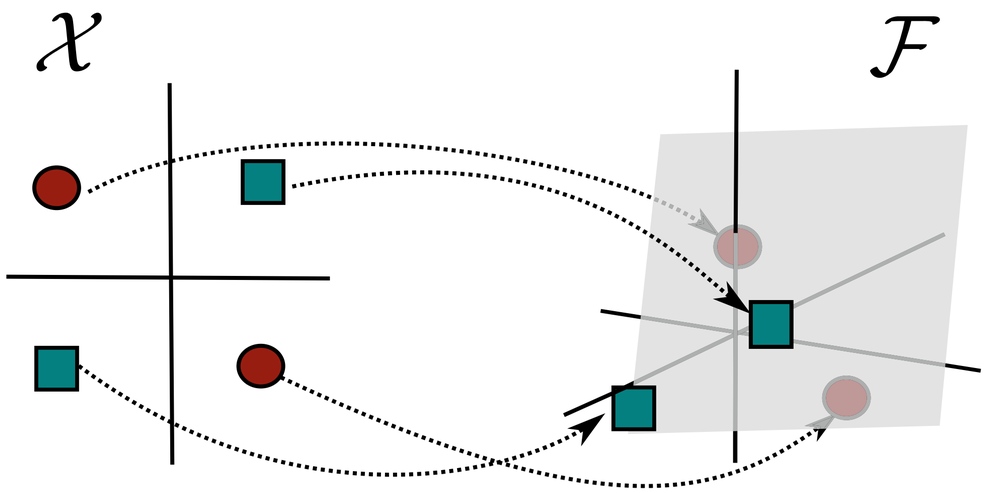
$$k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \,|\, \phi(\mathbf{z}) \rangle$$



$$\phi(x_1, x_2) = (x_1, x_2, 0.5(x_1^2 \times x_2^2))$$

WHY ARE KERNELS SO USEFUL?

Kernels are useful for **non-linearly separable data**; the feature map can be chosen such that the data is mapped to a higher dimensional feature space in which it becomes linearly separable.

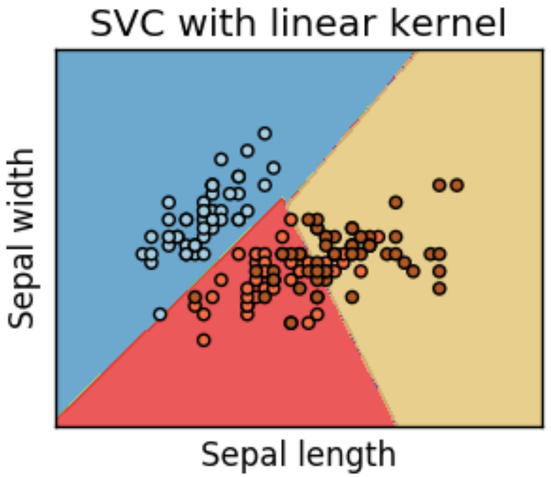


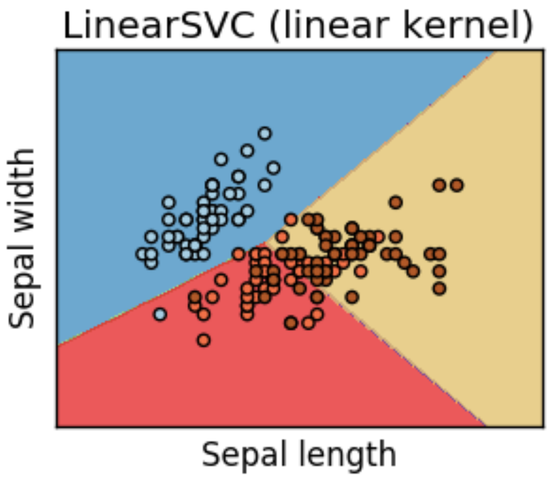
XOR Dataset

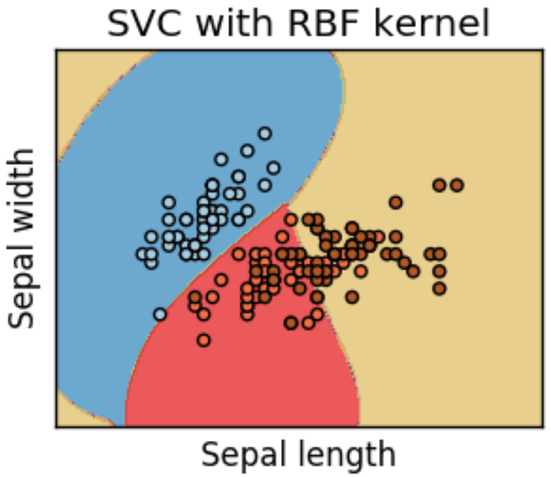
EXAMPLES OF KERNELS

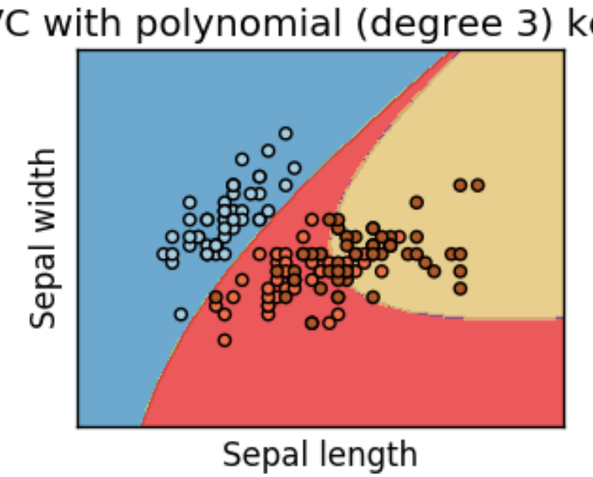
Name	Kernel	Hyperparameters
Linear	$k(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T\mathbf{z}$	–
Polynomial	$k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^T\mathbf{z} + c)^p$	$p \in \mathbb{N}, c \in \mathbb{R}$
Gaussian	$k(\mathbf{x}, \mathbf{z}) = e^{-\gamma\|\|\mathbf{x}-\mathbf{z}\|\|^2}$	$\gamma \in \mathbb{R}^+$

EXAMPLES OF KERNELS



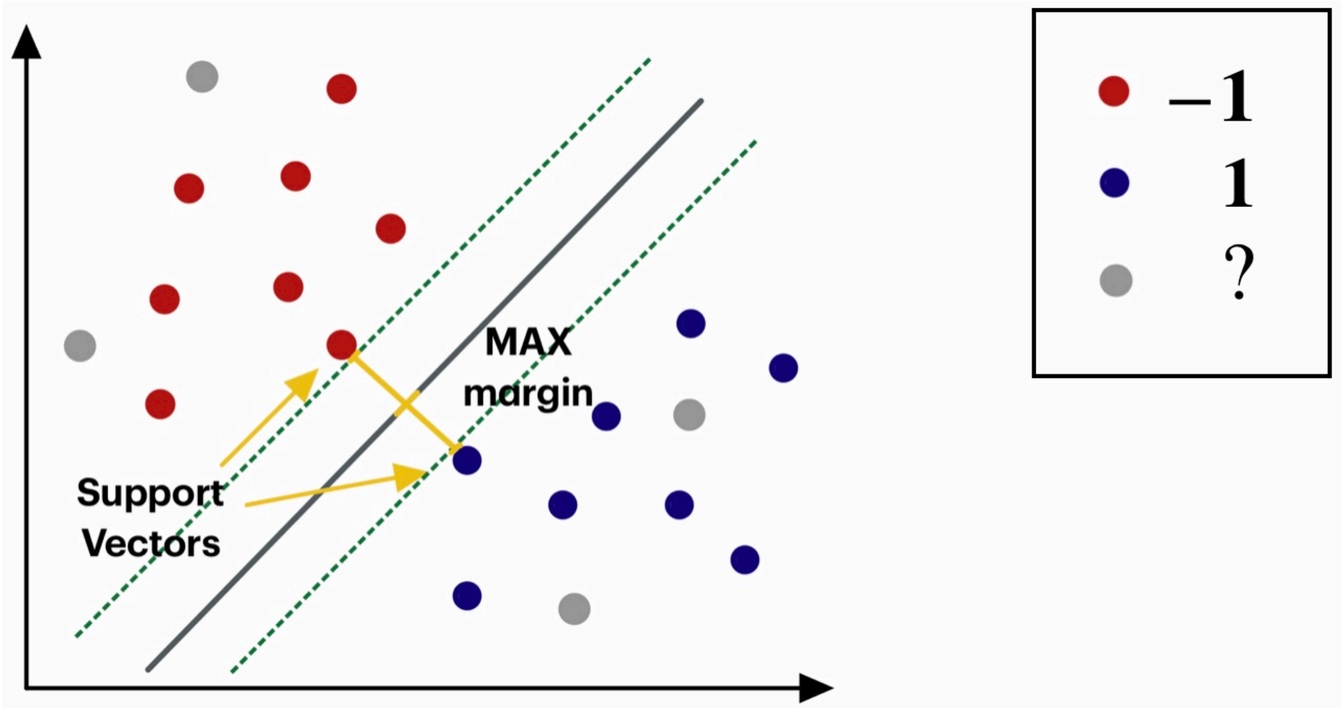






THE SUPPORT VECTOR MACHINE

Support Vector Machines try to **maximise the distance** between the closest training points (**the support vectors**) and the **separating hyperplane**.



THE SUPPORT VECTOR MACHINE

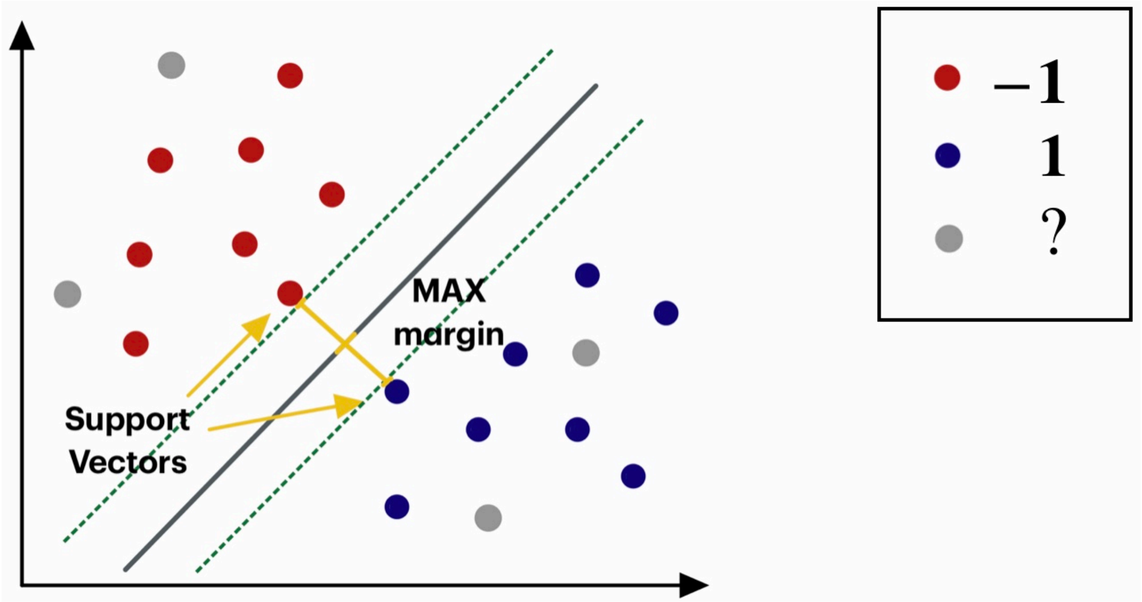
Support Vector Machines learn a model

$$f(\tilde{\mathbf{x}}) = \sum_{i=1}^{M} \alpha_i y_i k(\mathbf{x}^i, \tilde{\mathbf{x}}) + b$$

which is a linear model

$$f(\tilde{\mathbf{x}}) = \mathbf{w} \,|\, \phi(\tilde{\mathbf{x}})\rangle + b$$

where $\mathbf{w} = \sum_{i=1}^{M} \alpha_i y_i \langle \phi(\mathbf{x}^i) \,|$



QUANTUM KERNEL METHODS

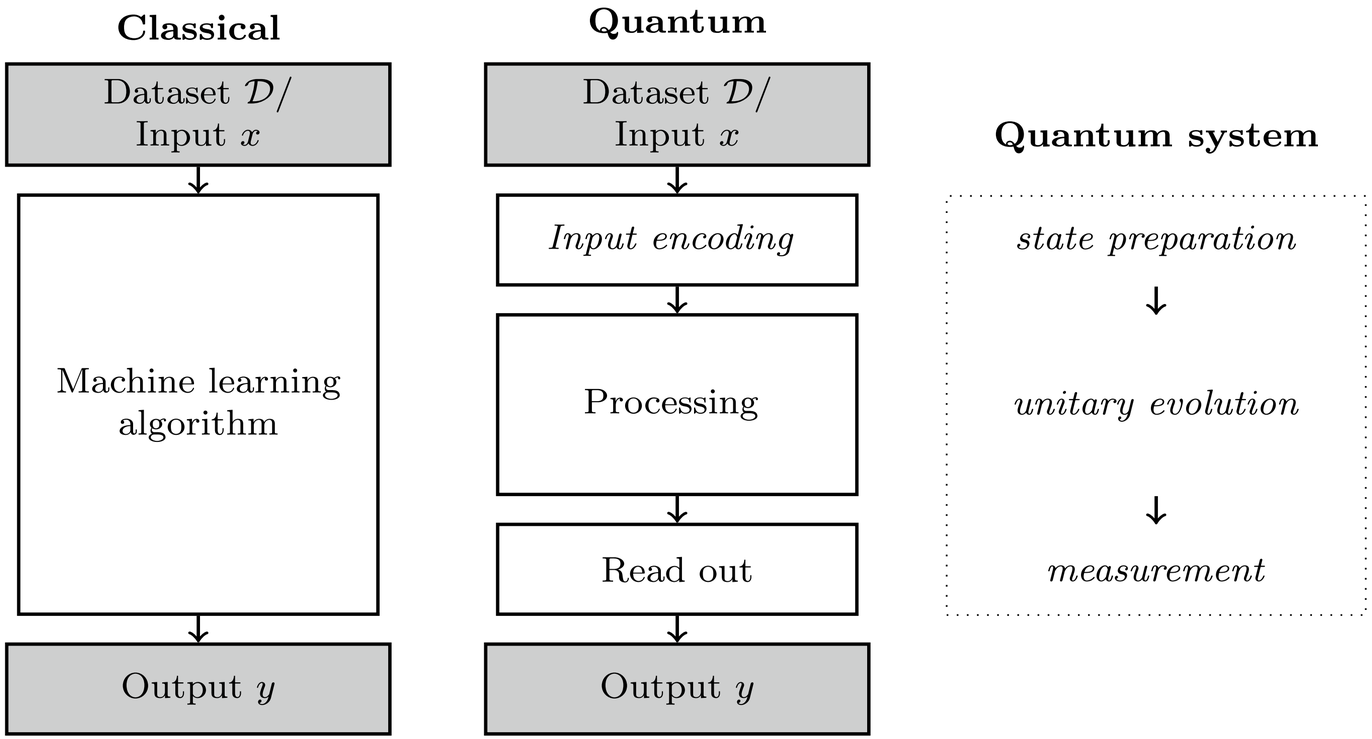
The computation of a quantum kernel involves **two key ideas**:

1. **Encoding** some classical data vector into a quantum state can be interpreted as **applying a feature map to that vector**.

2. The **measurement** of the squared state overlap between two quantum states encoding classical data **gives rise to a kernel**.

DATA ENCODING

If we want to process classical data on a quantum computer, we first have to decide **how to represent data by quantum states**. Data encoding is the first step in many quantum machine learning algorithms.



DATA ENCODING

The process of encoding inputs $\mathbf{x} \in X$ into a quantum system is, mathematically speaking, a **map** from the input space X to the state space of the quantum system.

We can interpret this as a **feature map**

$$\phi : \mathbf{x} \rightarrow |\phi(\mathbf{x})\rangle$$

This encoding will require some unitary operator $\mathcal{U}_\phi$ so we can write

$$|\phi(\mathbf{x})\rangle = U(\mathbf{x})\,|0\rangle^{\otimes n}$$

COMMON DATA ENCODING STRATEGIES: AMPLITUDE ENCODING

If $\mathbf{x} \in \mathbb{R}^n$ is a normalised classical data vector, meaning $\sum_k |x_k|^2 = 1$, then amplitude encoding gives rise to the following state

$$|\psi_{\mathbf{x}}\rangle = \sum_{i=1}^{2^n} x_i |i\rangle$$

Amplitude Encoding requires something called arbitrary state preparation. One algorithm for arbitrary state preparation is Möttonen State Preparation.

COMMON DATA ENCODING STRATEGIES: ROTATION ENCODING

For this encoding, the classical data vector $\mathbf{x} \in \mathbb{R}^n$ does not need to be normalised. However, you MAY want to limit $\mathbf{x} \in [0.2\pi]^{\otimes n}$. Rotation (Angle) Encoding with the R_y gate produces the following state

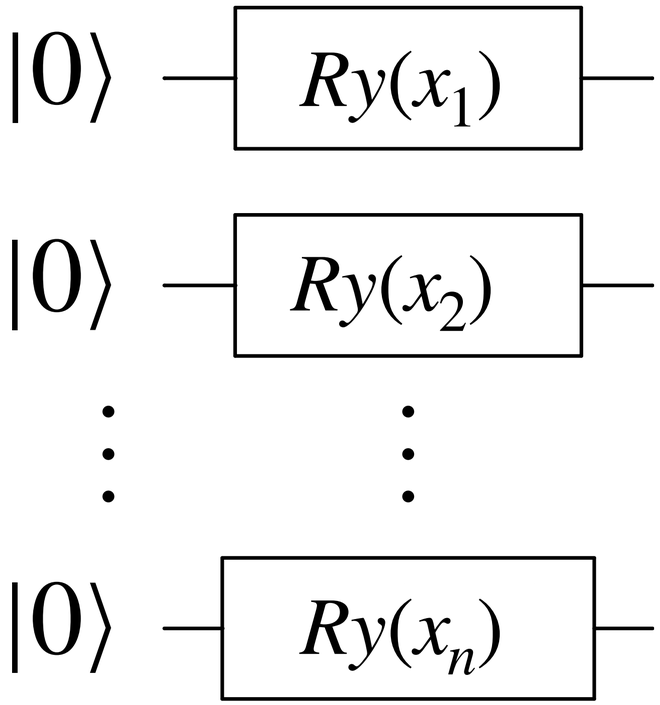
$$|\psi_{\mathbf{x}}\rangle = \sum_{q_1,\ldots q_n=0}^{1} \prod_{k=1}^{n} \cos(x_k)^{1-q_k} \sin(x_k)^{q_k} \, |q_1, \ldots q_n\rangle$$

COMMON DATA ENCODING STRATEGIES: ROTATION ENCODING

The unitary operator $U(\mathbf{x})$ required to encode the classical data vector with Rotation Encoding is

$$U(\mathbf{x}) = R_y(x_1) \otimes \ldots \otimes R_y(x_n)$$

This gives rise to a very simple circuit:



WHAT FEATURE MAPS DO THESE ENCODING STRATEGIES REPRESENT?

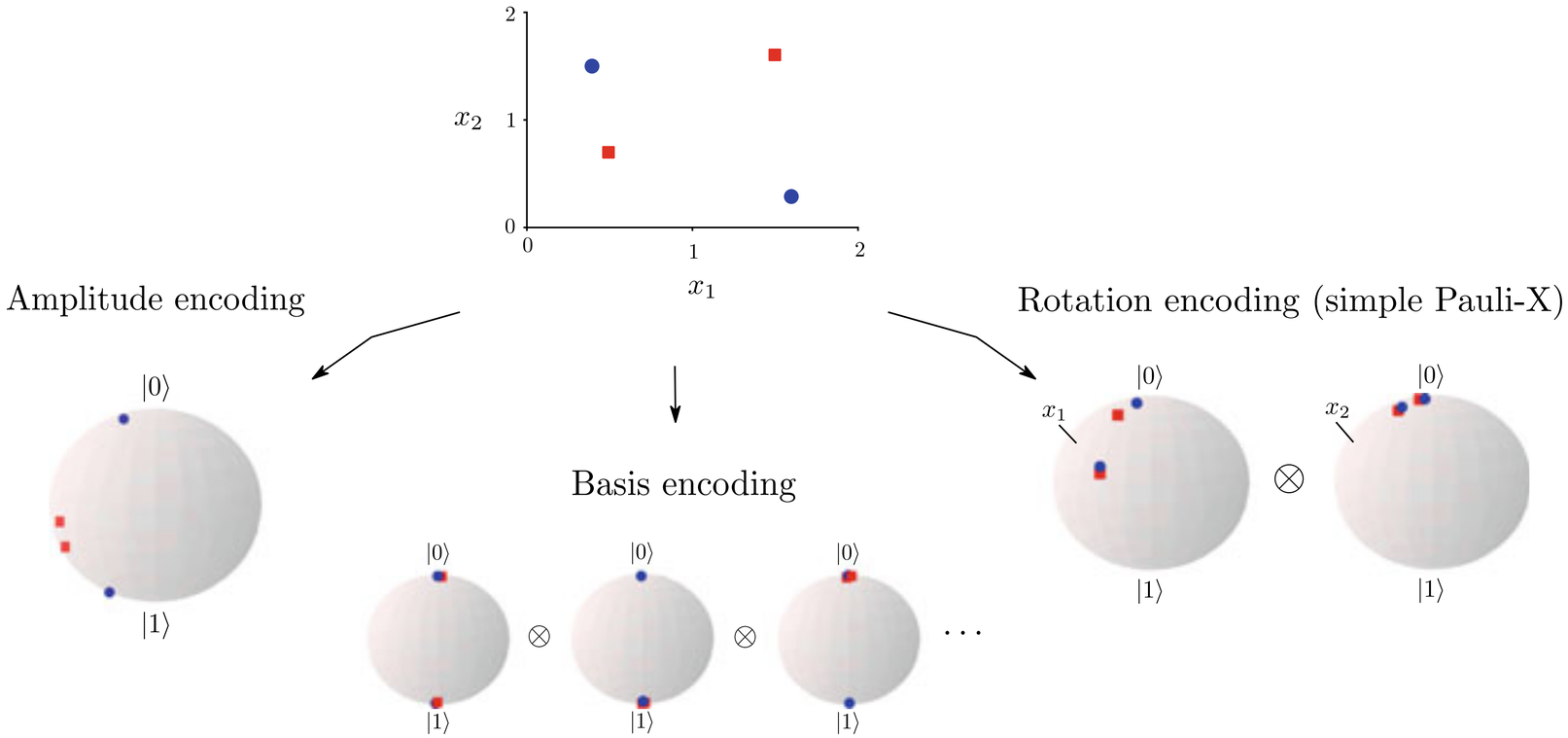
The feature map that amplitude encoding represents is

$$\mathbf{x} \rightarrow \sum_{i=1}^{2^n} x_i \, | \, i \rangle$$

The feature map that rotation encoding represents is

$$\mathbf{x} \rightarrow \sum_{q_1,\ldots q_n=0}^{1} \prod_{k=1}^{n} \cos(x_k)^{1-q_k} \sin(x_k)^{q_k} \, | \, q_1, \ldots q_n \rangle$$

WHAT FEATURE MAPS DO THESE ENCODING STRATEGIES REPRESENT?



QUANTUM KERNEL METHODS

The computation of a quantum kernel involves **two key ideas**:

1. **Encoding** some classical data vector into a quantum state can be interpreted as **applying a feature map to that vector**.

2. The **measurement** of the squared state overlap between two quantum states encoding classical data **gives rise to a kernel**.

ESTIMATING QUANTUM KERNELS

If $U(\mathbf{x})$ defines some unitary operation that encodes the classical datum $\mathbf{x}$ into an n-qubit quantum state as $|\phi(\mathbf{x})\rangle = U(\mathbf{x})\,|0\rangle^{\otimes n}$, then the kernel of two classical datapoints $\mathbf{x}$ and $\mathbf{z}$ is

$$k(\mathbf{x}, \mathbf{z}) = |\langle \phi(\mathbf{z}) \,|\, \phi(\mathbf{x}) \rangle|^2$$

The **squared state overlap then reveals the similarity** between the data in the feature space just as the inner product reveals this similarity for classical kernels.

$$k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \,|\, \phi(\mathbf{z}) \rangle$$

ESTIMATING QUANTUM KERNELS

If $U(\mathbf{x})$ defines some unitary operation that encodes the classical datum $\mathbf{x}$ into an n-qubit quantum state as $|\phi(\mathbf{x})\rangle = U(\mathbf{x})\,|0\rangle^{\otimes n}$, then the kernel of two classical datapoints $\mathbf{x}$ and $\mathbf{z}$ is

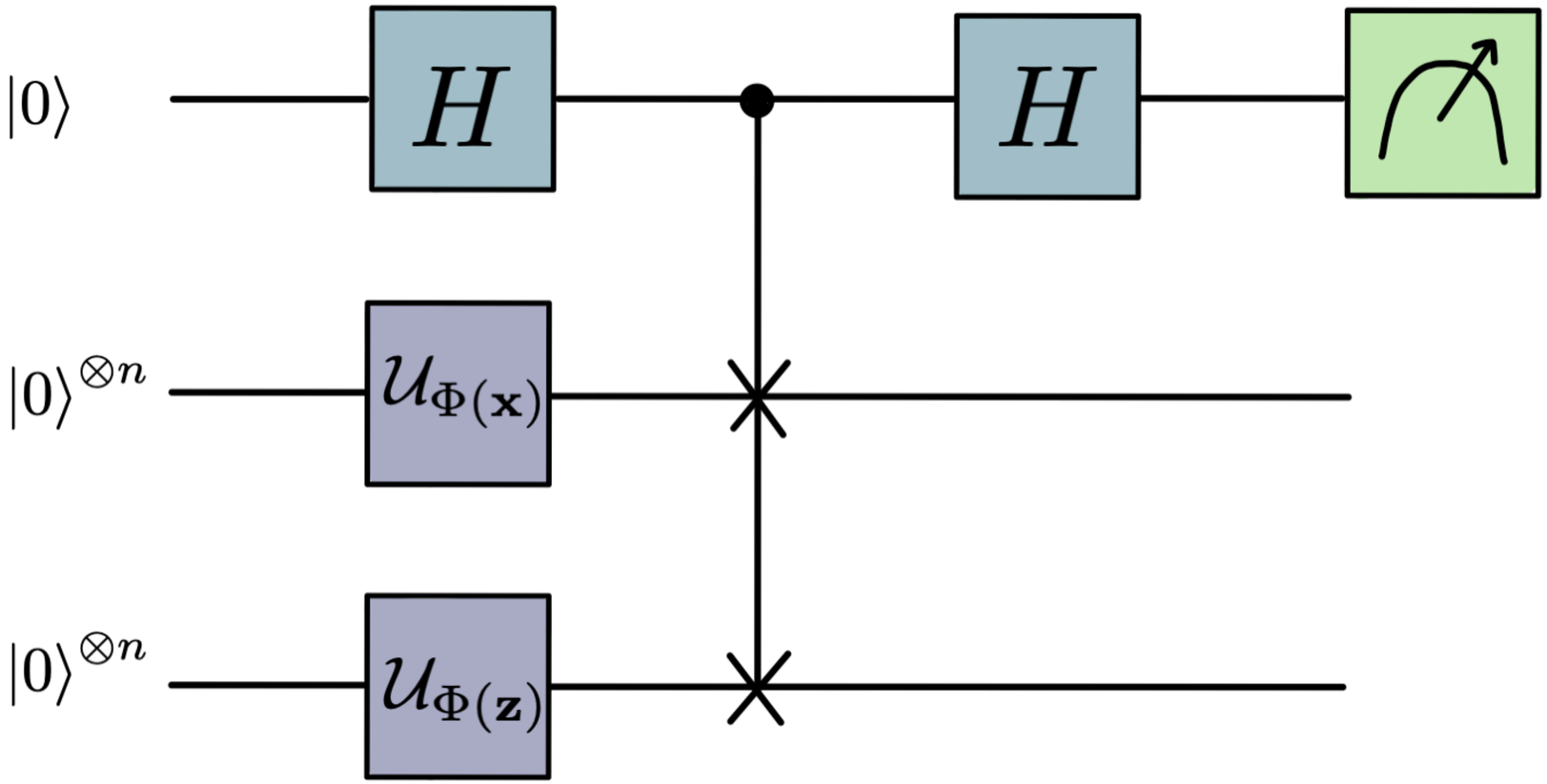
$$k(\mathbf{x}, \mathbf{z}) = |\langle \phi(\mathbf{z}) \,|\, \phi(\mathbf{x}) \rangle|^2$$

The **squared state overlap then reveals the similarity** between the data in the feature space just as the inner product reveals this similarity for classical kernels.

$$k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \,|\, \phi(\mathbf{z}) \rangle$$

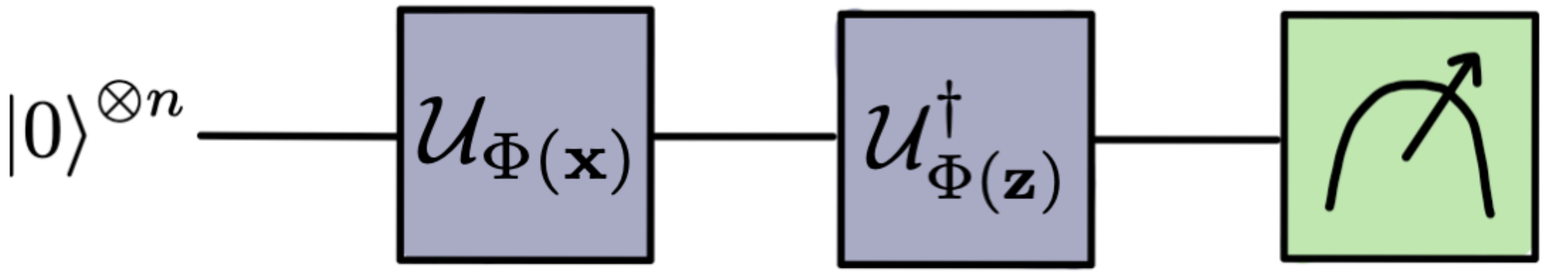
ESTIMATING QUANTUM KERNELS

The SWAP-Test

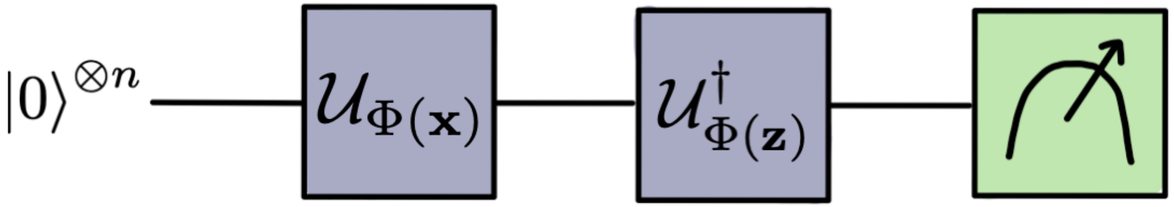


ESTIMATING QUANTUM KERNELS

The Inversion Test



THE INVERSION TEST



The Inversion Test is performed by preparing the state $U^{\dagger}_{\phi(\mathbf{z})} U_{\phi(\mathbf{x})} |0\rangle^{\otimes n}$

Thereafter, we need to measure the probability of measuring the qubits back in state $|0\rangle$

This measurement is defined by the projector $P = (|0\rangle\langle 0|)^{\otimes n}$

$$\begin{aligned}
\langle 0|^{\otimes n} U^{\dagger}_{\phi(\mathbf{x})} U_{\phi(\mathbf{z})} \; P \; U^{\dagger}_{\phi(\mathbf{z})} U_{\phi(\mathbf{x})} |0\rangle^{\otimes n} &= \left(\langle 0|^{\otimes n} U^{\dagger}_{\phi(\mathbf{x})} U_{\phi(\mathbf{z})} |0\rangle^{\otimes n}\right)\left(\langle 0|^{\otimes n} U^{\dagger}_{\phi(\mathbf{z})} U_{\phi(\mathbf{x})} |0\rangle^{\otimes n}\right) \\
&= |\langle 0|^{\otimes n} U^{\dagger}_{\phi(\mathbf{z})} U_{\phi(\mathbf{x})} |0\rangle^{\otimes n}|^2 \\
&= |\langle \phi(\mathbf{z}) | \phi(\mathbf{z}) \rangle|^2 .
\end{aligned}$$

It can be seen that this measurement returns the quantum kernel $k(\mathbf{x}, \mathbf{z}) = |\langle \phi(\mathbf{z}) | \phi(\mathbf{x}) \rangle|^2$

EXAMPLES OF QUANTUM KERNELS

Interestingly, the standard data encoding strategies used in quantum machine learning give rise to kernels that resemble classical kernels.

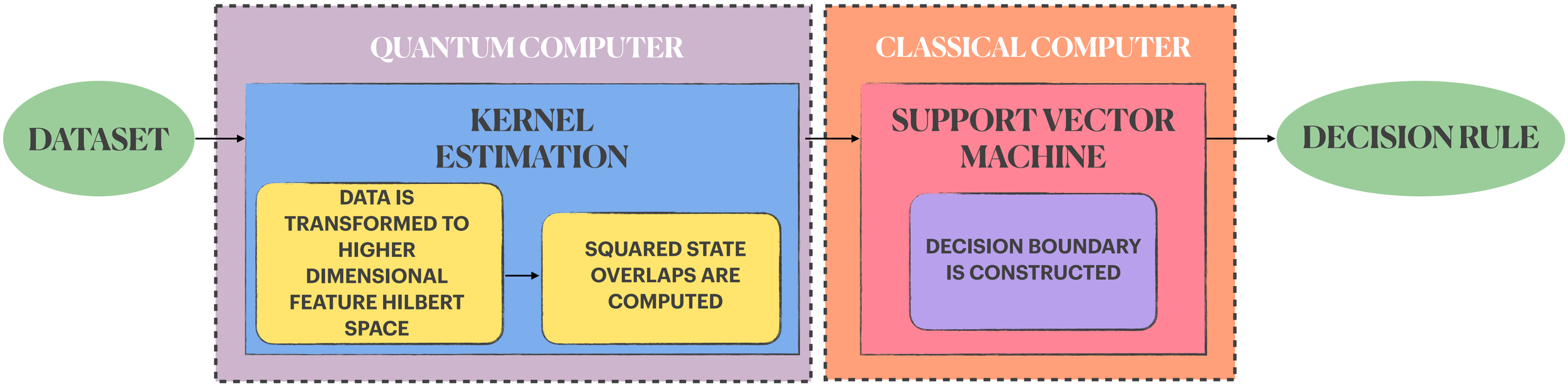
The kernel that arises from amplitude encoding is the absolute square of the classical linear kernel.

$$k(\mathbf{x}, \mathbf{z}) = |\langle \mathbf{x} | \mathbf{z} \rangle|^2$$

The kernel that arises from rotation encoding is related to the classical cosine kernel.

$$k(\mathbf{x}, \mathbf{z}) = \prod_{k=1}^{n} |\cos(\mathrm{x_k} - \mathrm{z_k})|^2$$

THE QUANTUM KERNEL-BASED CLASSIFIER



THANK YOU FOR LISTENING