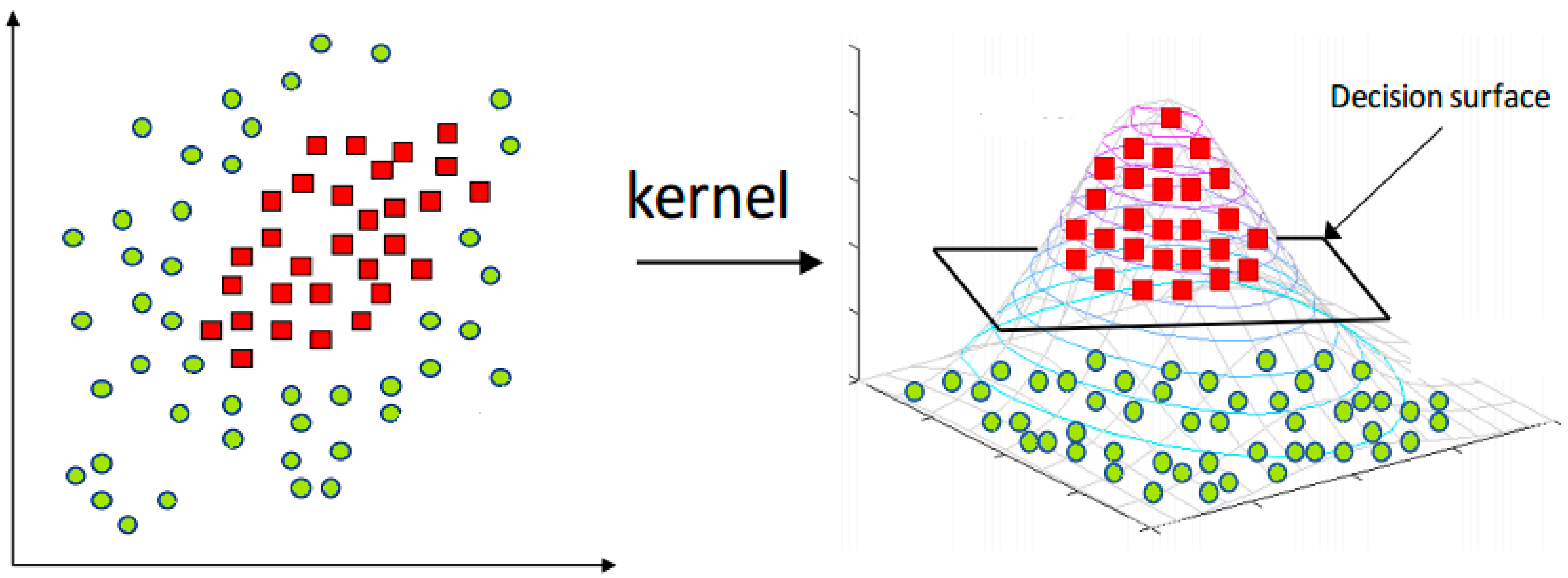
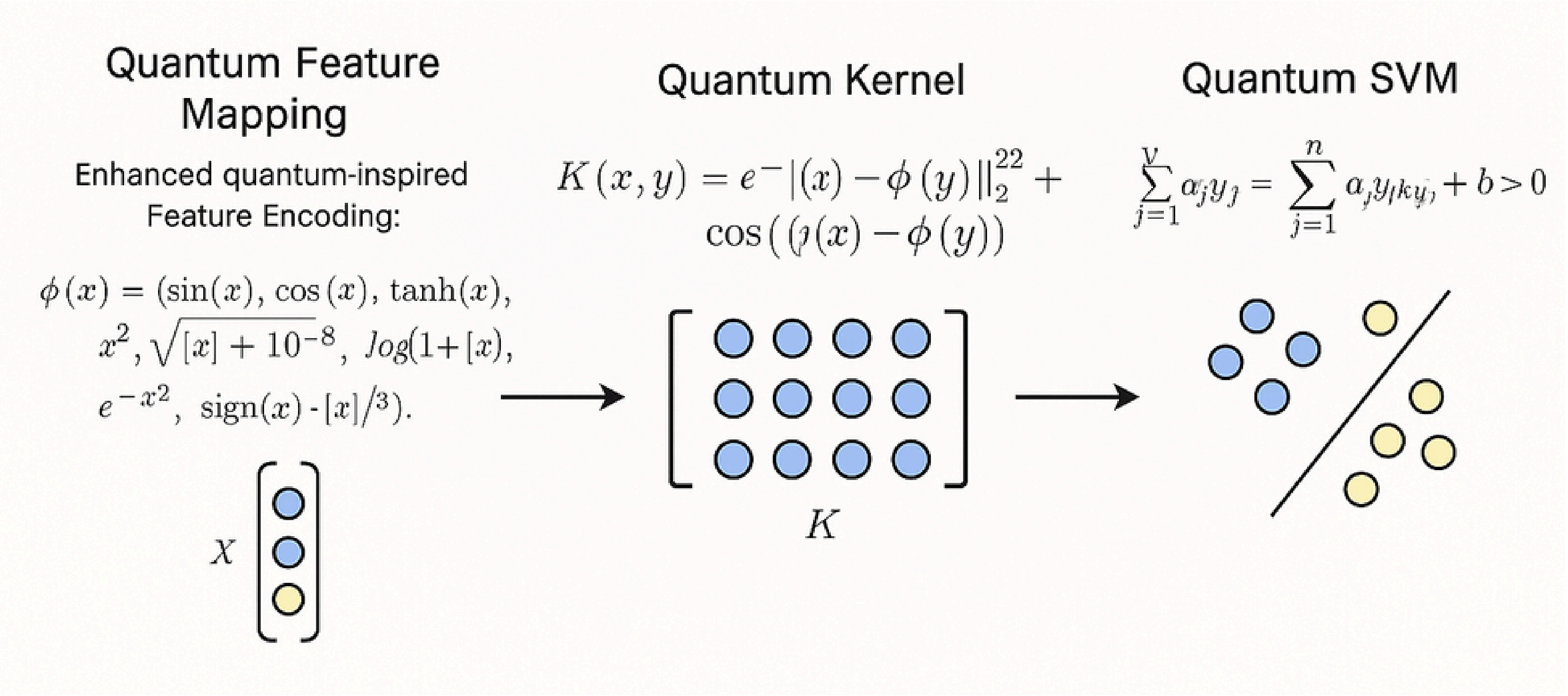
QSVM

Classical SVM

Quantum Feature Mapping QSVM





https://www.linkedin.com/pulse/role-svm-model-current-data-science-deepak-kumar/

QSVM Mathematics

Quantum Feature Mapping:

Transforms classical features into a richer quantum-like space using a combination of trigonometric, exponential, and nonlinear functions to enhance pattern separability.

$$\phi(x) = \left[\sin(x),\ \cos(x),\ \tanh(x),\ x^2,\ |x|,\ \log(1+|x|),\ e^{-x^2},\ \mathrm{sign}(x)\cdot|x|^{1/3}\right]$$

Quantum Kernel Function:

Measures similarity between quantum-transformed data points using both Gaussian (RBF) and cosine components to capture both local and global feature relationships.

$$K(x_i, x_j) = \begin{cases} 2, & \text{if } \|x_i - x_j\| = 0 \\ \exp(-\gamma \cdot \|x_i - x_j\|^2) + \cos(\|x_i - x_j\|), & \text{if } \|x_i - x_j\| > 0 \end{cases}$$

- γ is the kernel sensitivity hyperparameter.
- ||.|| is the Euclidean distance.

SVM Classifier with Quantum Kernel:

Performs classification by finding the optimal decision boundary in the quantum-enhanced space using support vectors and the custom kernel function.

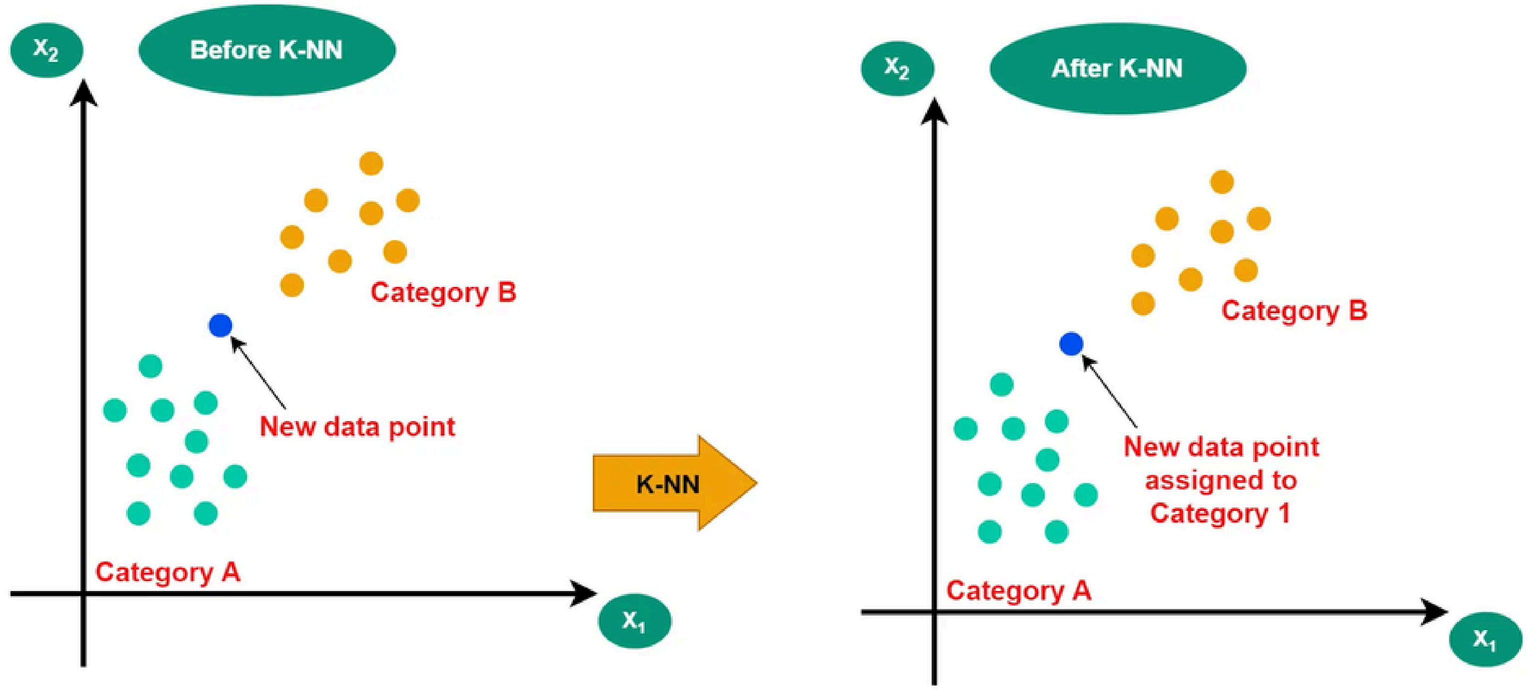
$$f(\mathbf{x}) = \mathrm{sign}\left(\sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right)$$

- α_i are the learned Lagrange multipliers
- $y_i \in \{-1,+1\}$
- b is the bias term
- $K(\cdot,\cdot)$ is the quantum kernel

Source: https://medium.com/mit-6-s089-intro-to-quantum-computing/quantum-support-vector-machine-qsvm-134eff6c9d3b

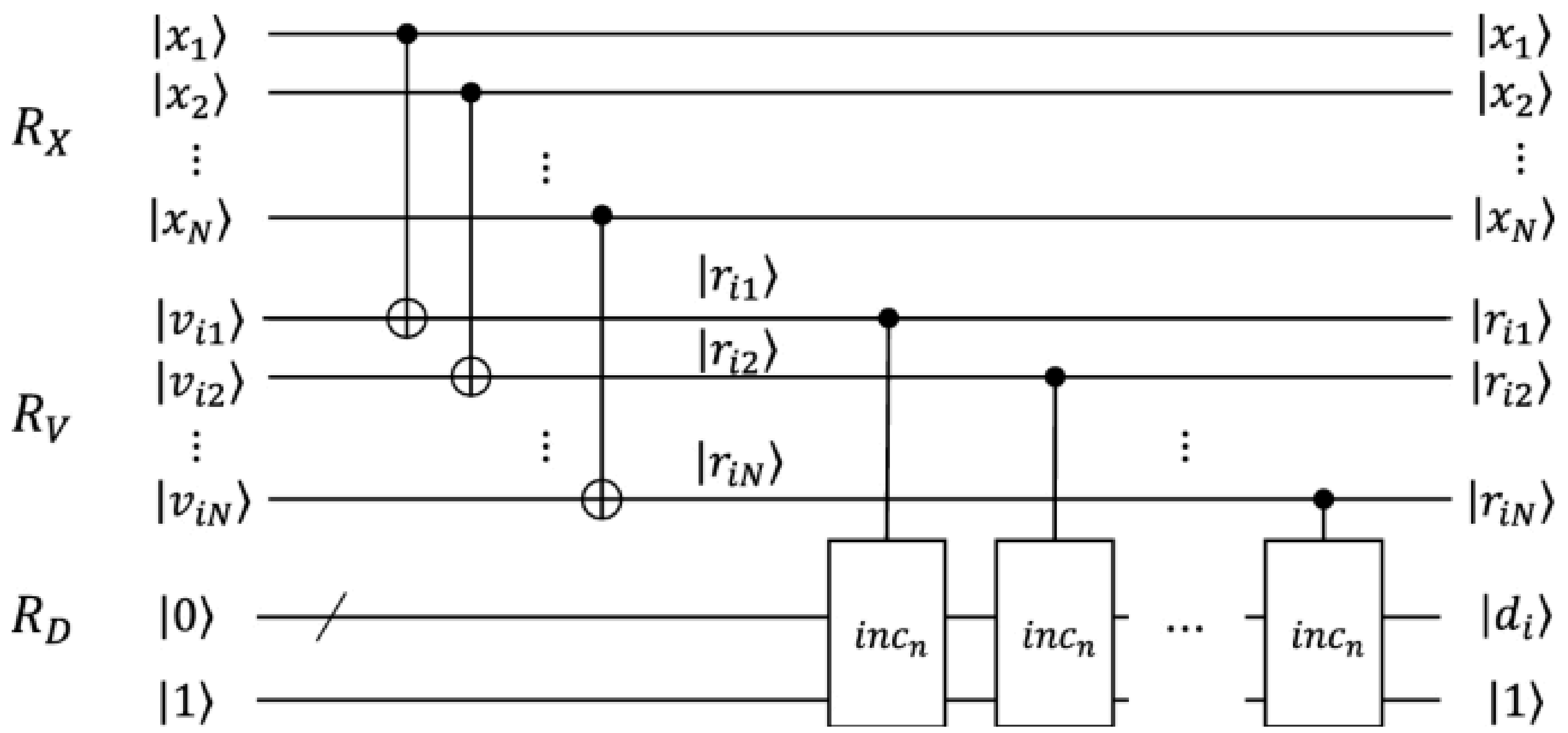
QKNN

Classical KNN



https://medium.com/@sahin.samia/demystifying-k-neighbors-classifier-knn-theory-and-python-implementation-from-scratch-f5e76d6f2d48

Quantum K Nearest Neighbor QKNN



https://link.springer.com/article/10.1007/s11128-021-03361-0

QKNN Mathematics

Quantum Kernel Matrix Computation:

This quantum kernel $k(x_i, x_j)$ computes the similarity between quantum-encoded data points x_i and x_j via inner product in Hilbert space. It replaces classical distance-based similarity in KNN with a quantum feature map ϕ(·), enabling the exploitation of quantum state overlaps for classification.

$$K_{ij} = \kappa(x_i, x_j) = \left|\langle \phi(x_i) | \phi(x_j) \rangle\right|^2$$

Euclidean Distance in Quantum Feature Space: (SWAP Test)

Even though the data is transformed into a quantum feature space, KNN still uses Euclidean distance on the resulting kernel matrix to find the nearest neighbors. This distance $d(x_i, x_j)$ determines similarity between a test point x and training point x_i .

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{k=1}^{n} (x_i - x_j)^2}$$

Prediction via Majority Voting:

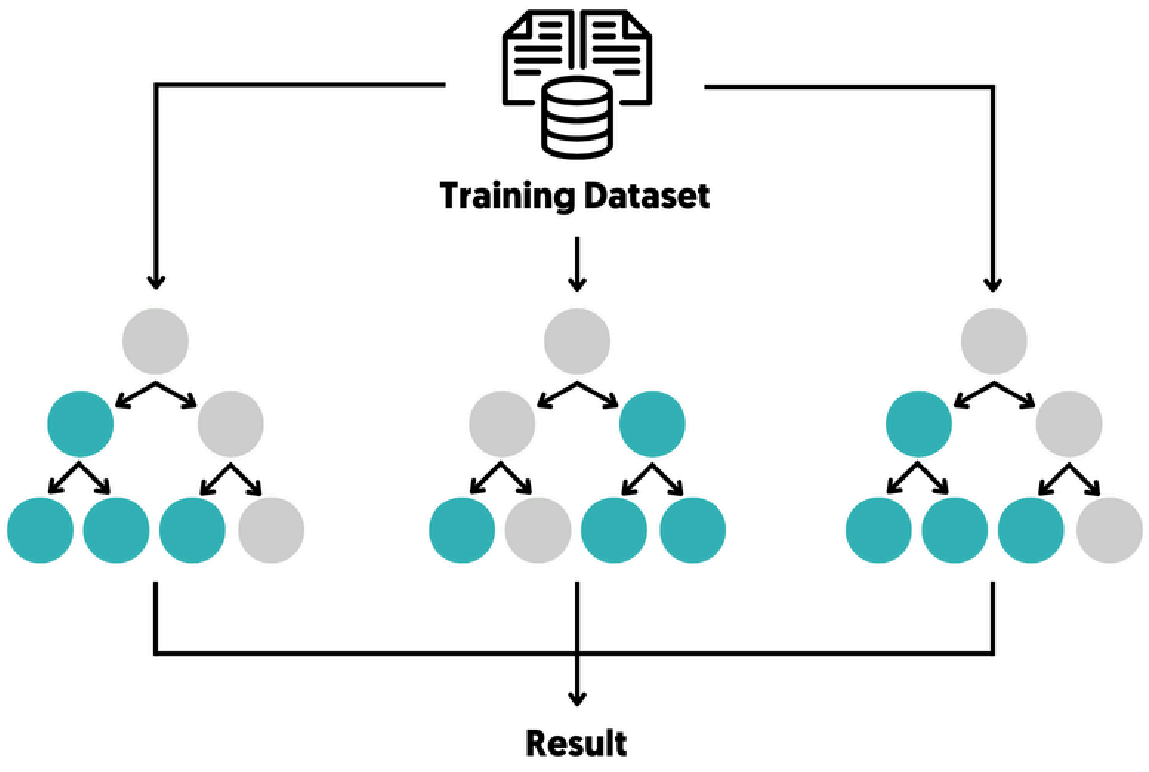
Once the k nearest neighbors $N_k(x)$ are identified using quantum kernel distances, the final predicted class y^ is selected via majority voting among the labels y_i of those neighbors.

$$\hat{y} = \mathbf{mode}\left(y_i \mid \mathbf{x}_i \in \mathcal{N}_k(\mathbf{x})\right)$$

Source: https://www.mdpi.com/2227-7390/12/12/1872#:~:text=The%20time%20complexity%20of%20the,up%20for%20large%20vector%20dimensions.

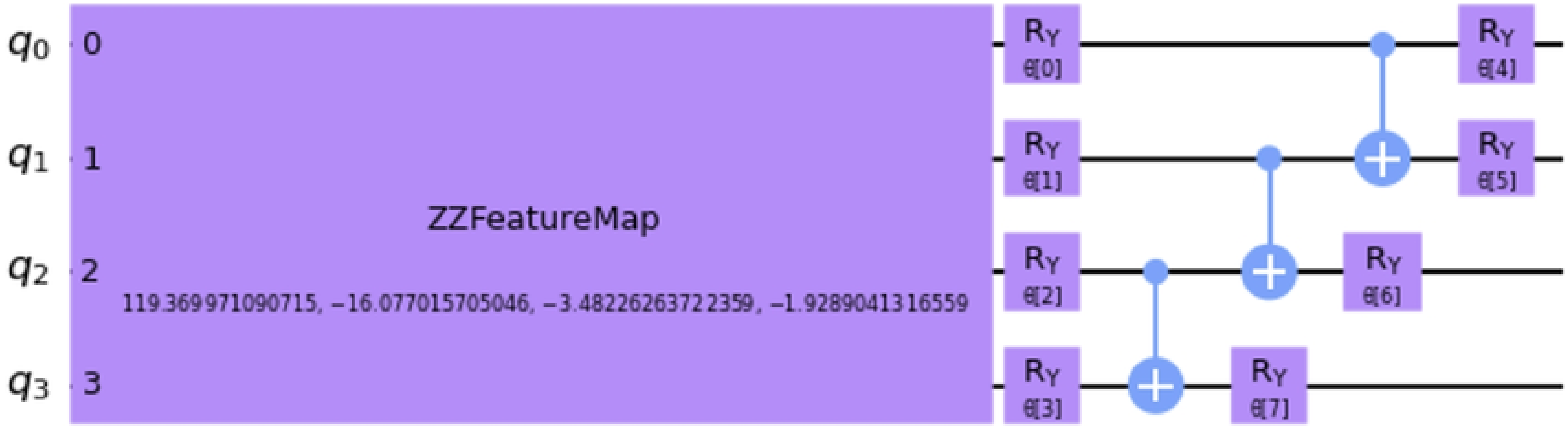
QRF

Classical QRF



https://dida.do/what-is-random-forest

Quantum Random Forest QRF



https://www.researchgate.net/figure/Quantum-random-forest-circuit-It-consists-of-a-feature-map-that-encodes-the-input-data_fig6_380293282

QRF Mathematics

Quantum Feature Map Transformation

A quantum feature map ϕ(x) encodes classical input $x \in R^n$ into a quantum state via a parameterized unitary U(x). The resulting quantum state ϕ(x) captures nonlinear correlations, enabling enhanced pattern recognition when used in classical models like Random Forest.

$$\phi : \mathbb{R}^n \to \mathbb{C}^{2^m}, \quad \phi(x) = U_\theta(x) \left|0\right\rangle^{\otimes m}$$

Prediction via Majority Voting in Quantum Forest

Each decision tree h_t in the quantum-enhanced forest predicts a class label using the quantum-transformed input ϕ(x). The final prediction y^ is the majority vote across T such trees—retaining classical ensemble logic while operating in a quantum-enhanced feature space.

$$\hat{y} = \mathrm{mode}\left(\{h_t(\phi(x))\}_{t=1}^{T}\right)$$

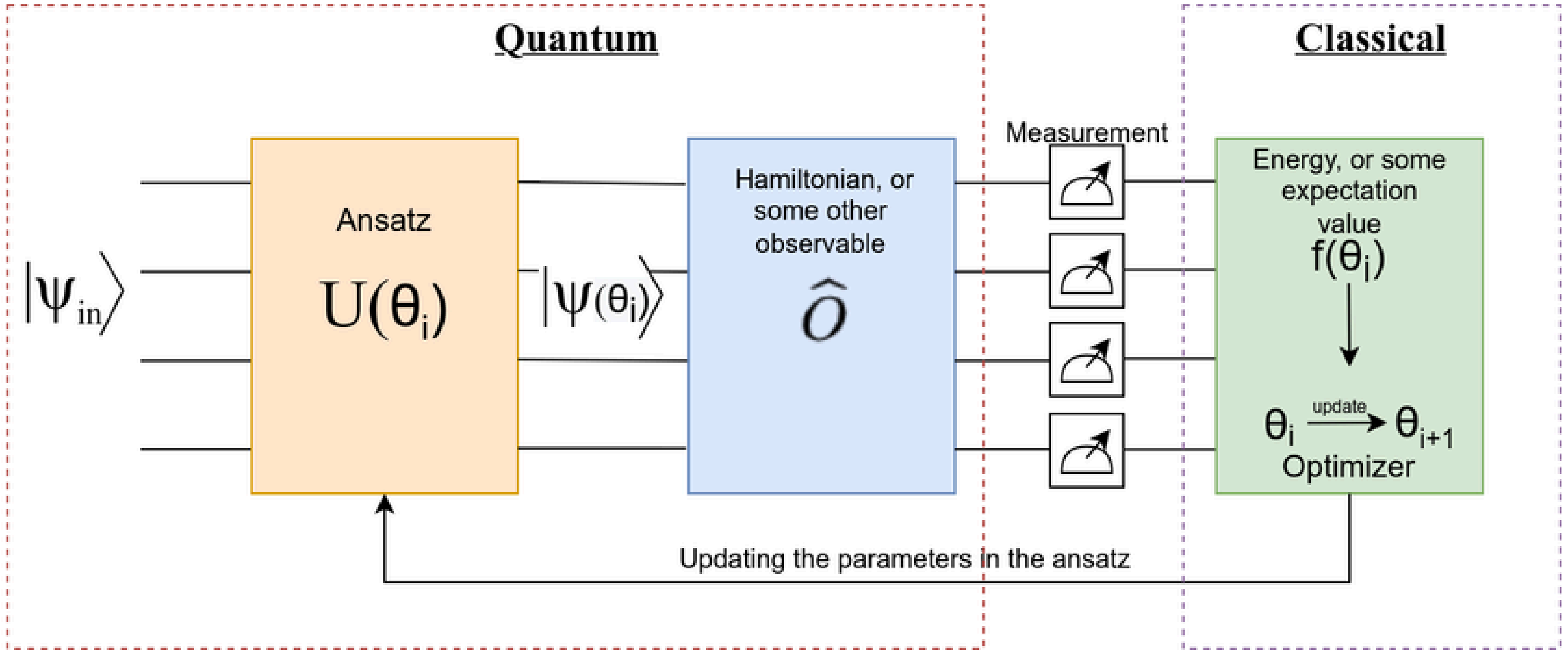
Tree-Level Decision Rule

Each decision node in a tree h_t evaluates a single quantum-transformed feature $\phi(x)_j$ against a threshold τ. The quantum feature values guide the tree's branching logic, enabling better class separation with fewer trees or shallower depth than classical RF in some cases.

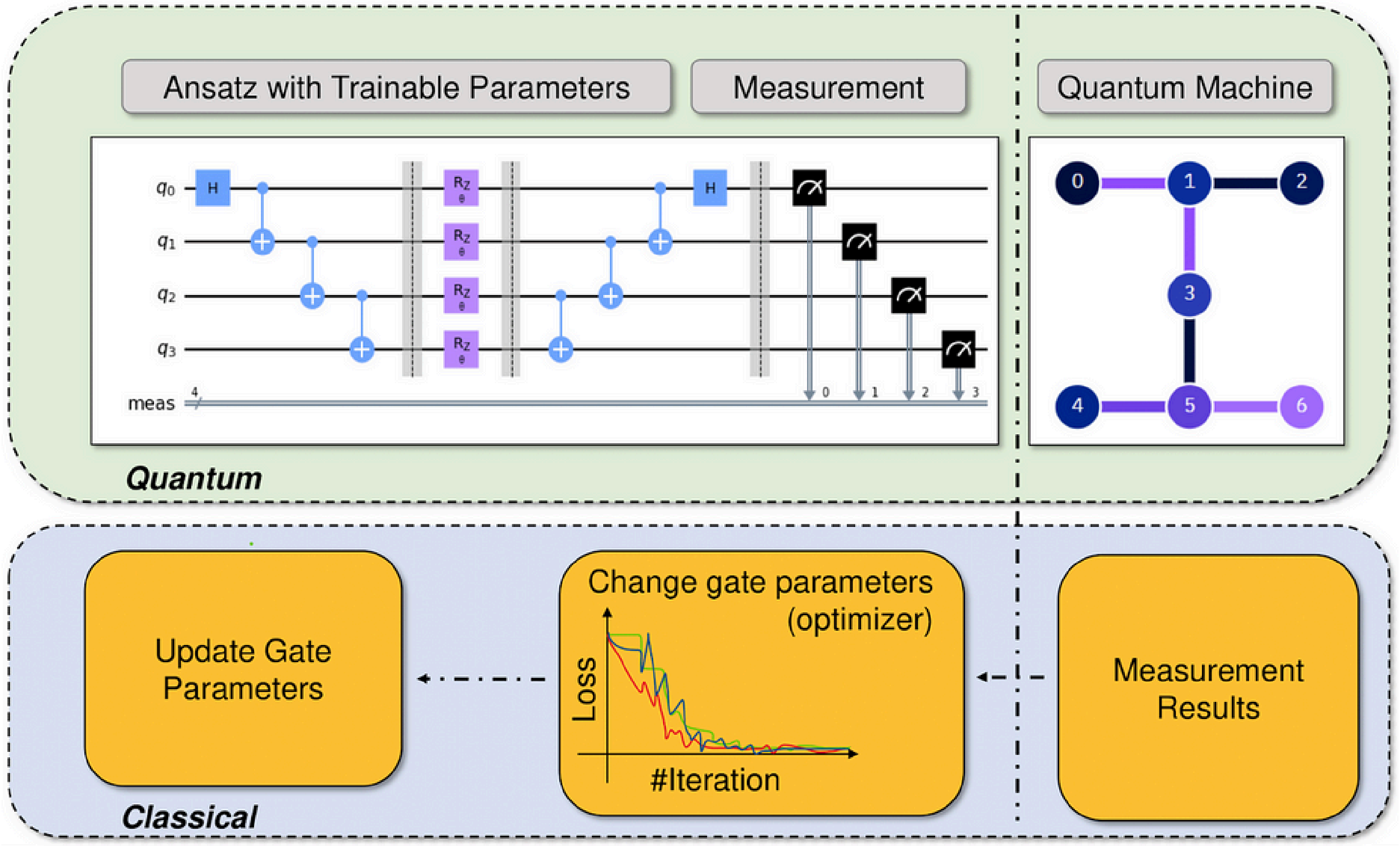
$$h_t(\phi(x)) = \begin{cases} 1, & \text{if } \phi(x)_j \le \tau \\ 0, & \text{otherwise} \end{cases}$$

Source : https://ceur-ws.org/Vol-2842/paper_3.pdf

QNN/ VQC



https://www.researchgate.net/figure/Basic-Structure-of-a-Variational-Quantum-Algorithm-The-algorithm-starts-with-an-initial_fig1_362859187



https://medium.com/qiskit/enhance-variational-quantum-algorithms-with-qiskit-pulse-and-qiskit-dynamics-768249daf8dd

QNN/ VQC Mathematics

Quantum Encoding of Input

The input vector $x \in R_n$ is encoded into a quantum state using a data-encoding unitary U_{enc}. This state $|\psi(x)\rangle\rangle$ serves as the starting point for quantum processing in the VQC model.

$$|\psi(x)\rangle = U_{\text{enc}}(x)|0\rangle^{\otimes n}$$

Parametrized Variational Circuit

A trainable quantum circuit $U_{var}(\theta)$, composed of parameterized quantum gates, transforms the encoded state. The parameters θ are optimized during training to minimize classification error, similar to weights in neural networks.

$$|\psi_\theta(x)\rangle = U_{\text{var}}(\theta)\,|\psi(x)\rangle = U_{\text{var}}(\theta)U_{\text{enc}}(x)|0\rangle^{\otimes n}$$

Final Prediction via Expectation Measurement

Prediction is made by measuring an observable M (e.g., a Pauli operator) on the output state. The expectation value determines the class label, often mapped using a threshold or sign function for binary classification.

$$\hat{y}\,(x) = sign(\langle\psi_\theta(x)|M|\psi_\theta(x)\rangle)$$

Source: https://www.nature.com/articles/s41467-023-36144-5