

Machine Learning - Set of all tasks in which a computer can make decisions based on data

It should be able to:

Remember: look at a dataset

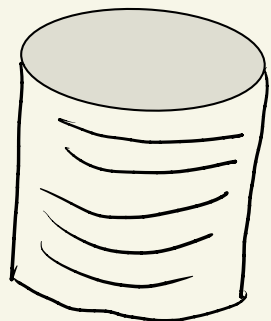
Formulate: Model (Rule) based on dataset

Predict: Make predictions on new data
outside dataset

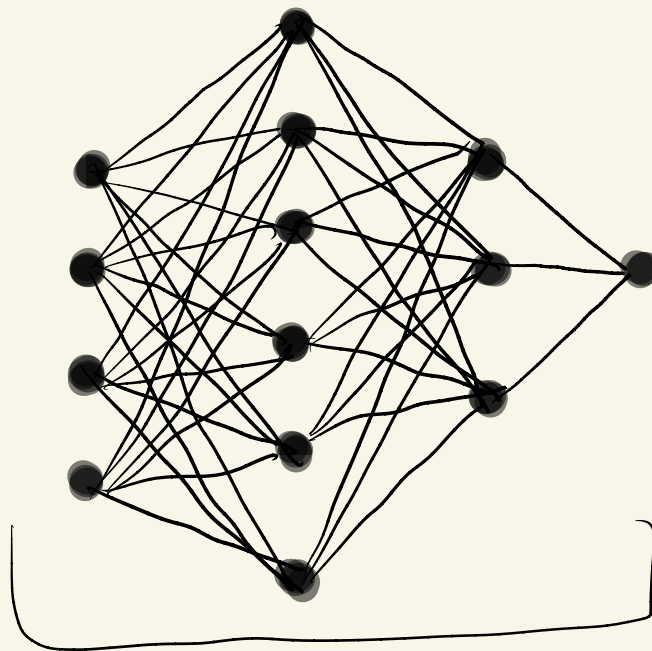
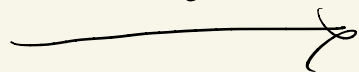
There are many flavors of ML

- supervised / unsupervised
- semi supervised
- reinforcement learning
- Generative learning

database



image

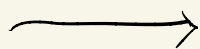
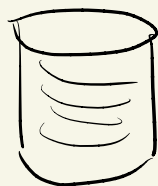


cat / dog

Images
of
cats / dogs

M

new data



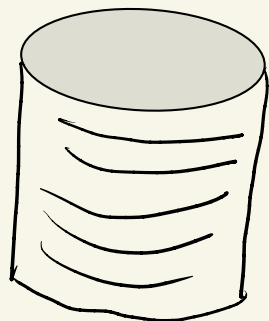
M



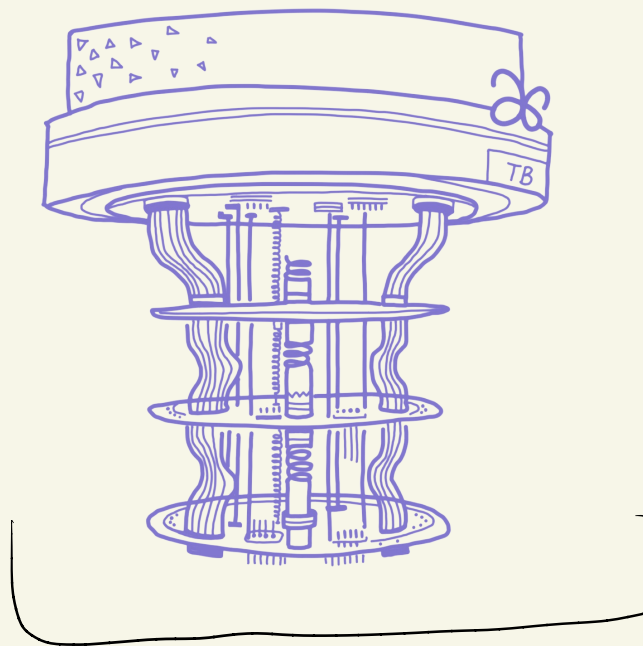
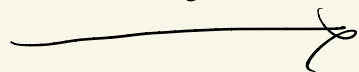
cat / dog

generalization / overfitting

database



image

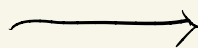


→ cat / dog

Images
of
cats / dogs

M

new data

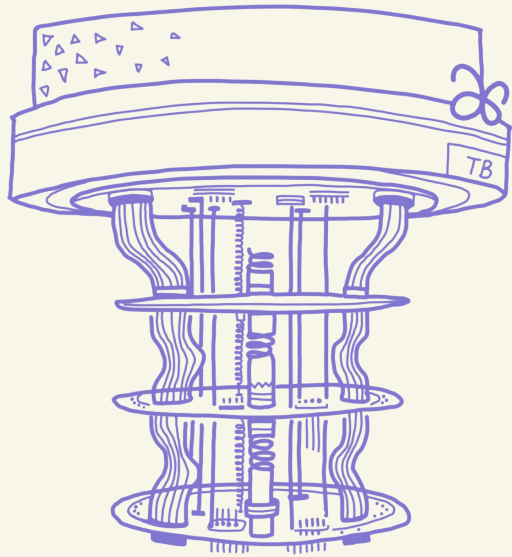


M

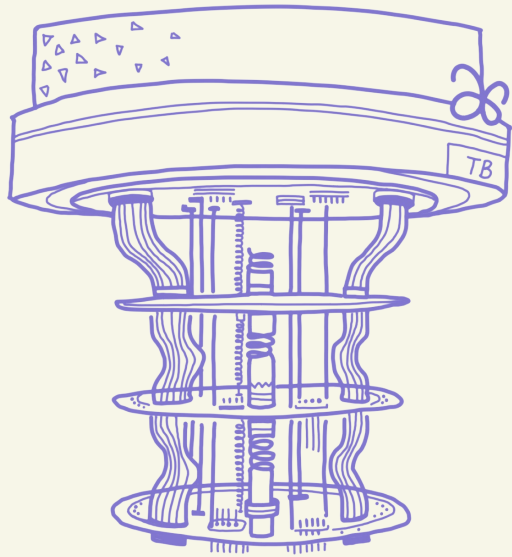


cat / dog

generalization / overfitting

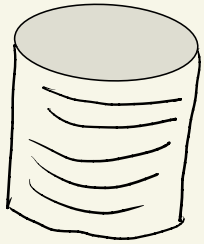


- encodes the data
- processes the data (somehow)
- Returns prediction



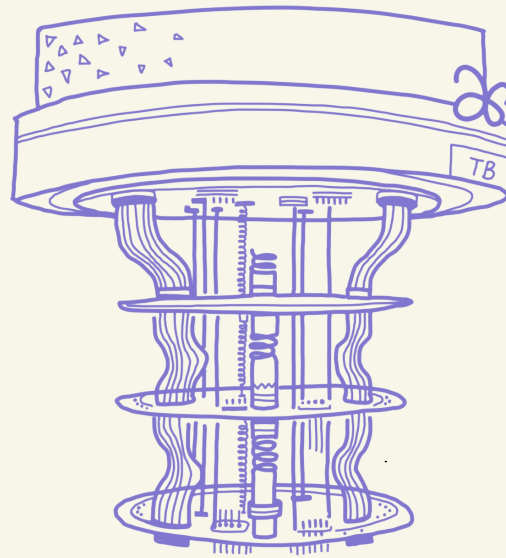
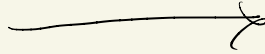
- encodes the data 
- processes the data (somehow)
- Returns prediction

database



Images
of
cats/dogs

image



Kernel method

Parameterized Quantum
circuit

Support vector machine

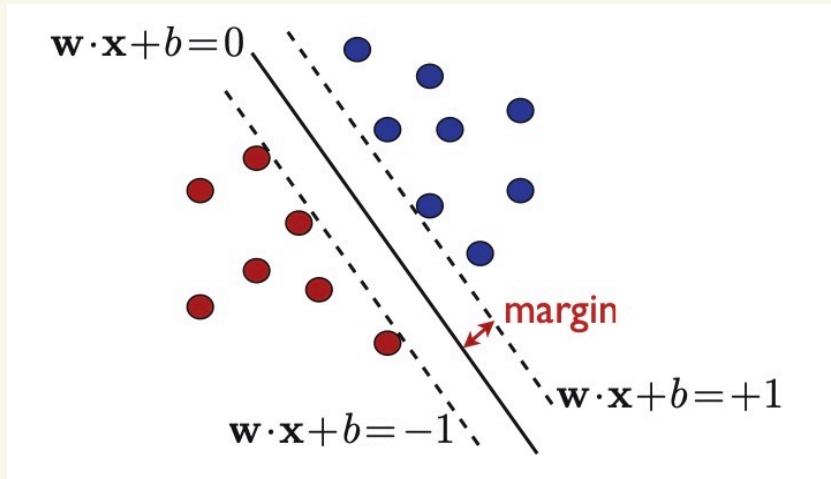
a.k.a Quantum Neural Network

Kernel ridge regression / Gaussian processes
kernel PCA

classification / regression / generator

Support Vector Machine (Supervised setting)

Dataset with M datapoints (x_i, y_i)



$$x_i \mapsto \text{sign}(w \cdot x + b)$$

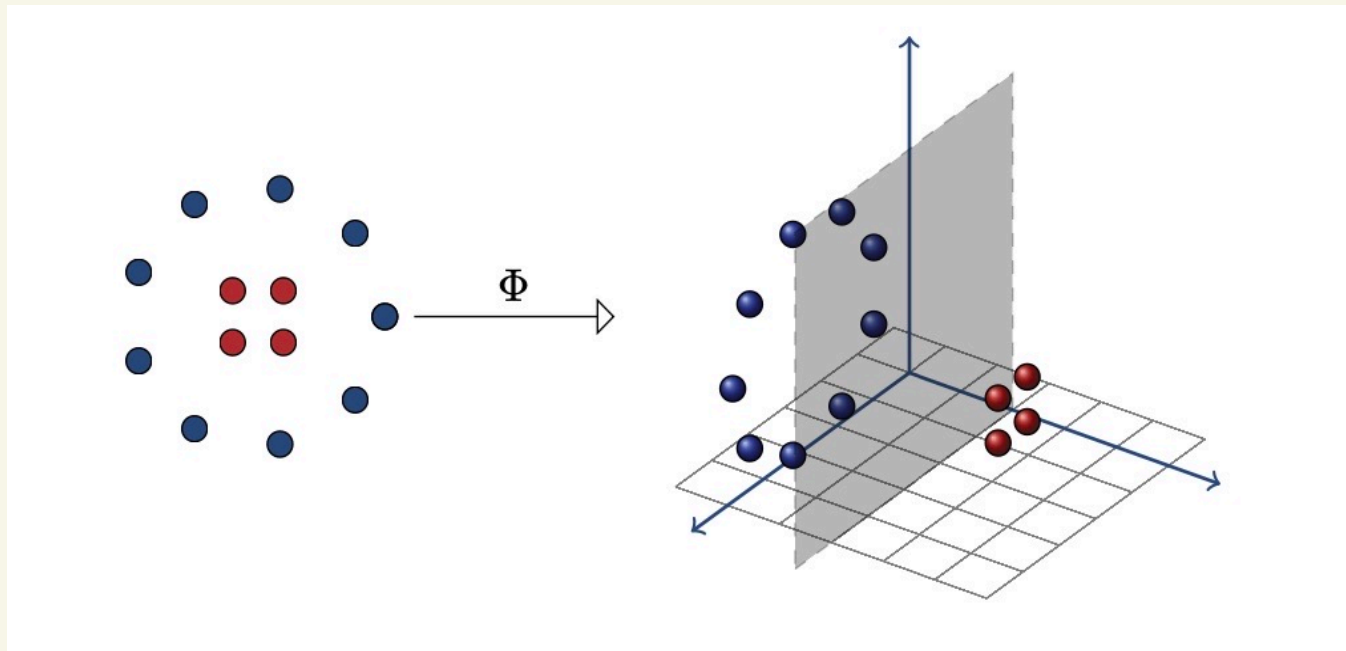
Dual Formulation:

$$L = \sum_{i=0}^{M-1} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i \cdot x_j$$

$$x_i \mapsto \text{sign} \left(\sum_{i=0}^{M-1} \alpha_i y_i (x_i \cdot x) + b \right)$$

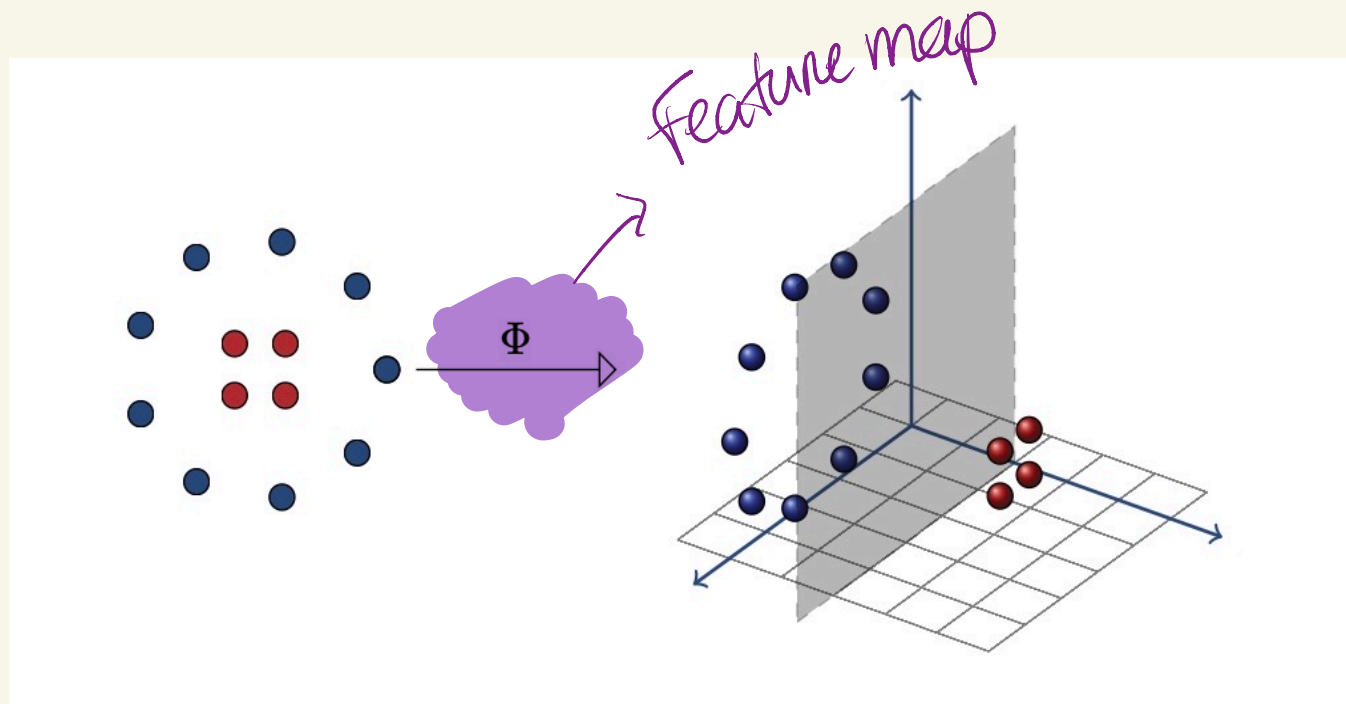
Problem : Data might not be linearly separable

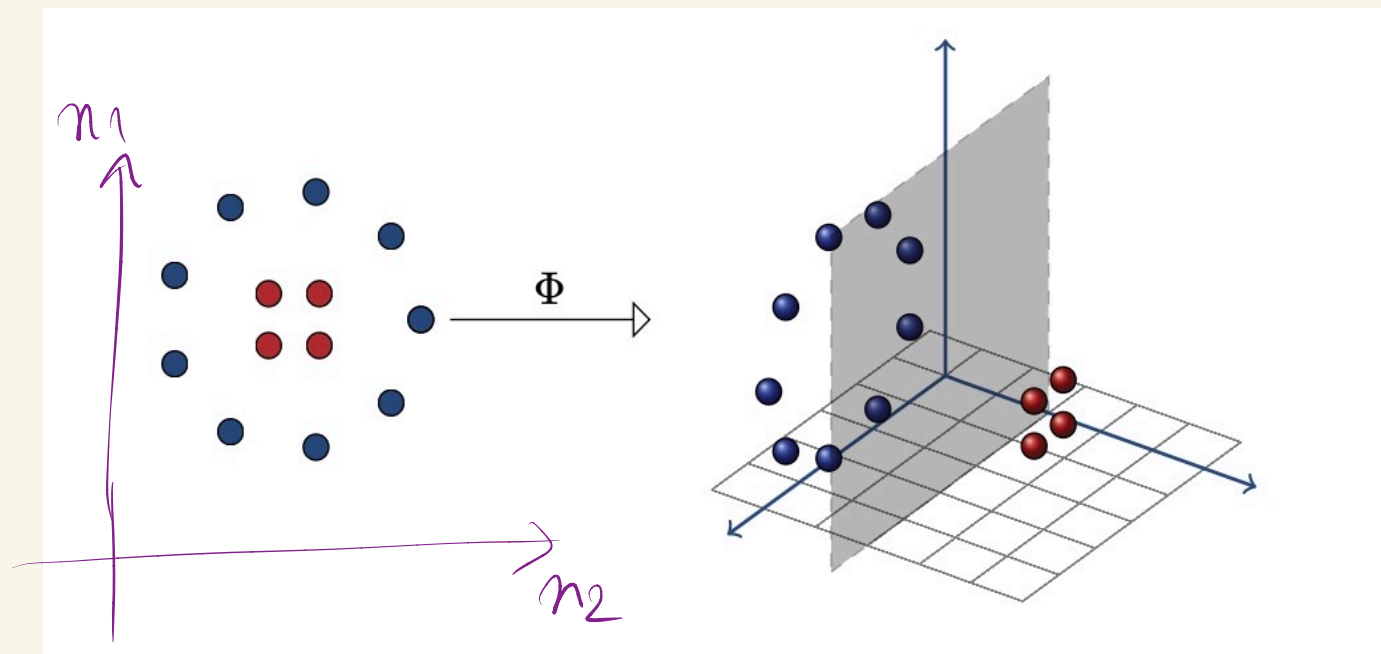
Solution : map data to higher dimension



Problem : Data might not be linearly separable

Solution : map data to higher dimension





$$\phi : (n_1, n_2) \mapsto (n_1, n_2, (n_1^2 + n_2^2))$$

Support vector machine can

estimate inner products in ϕ space

$$L = \sum_{i=0}^{m-1} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j \gamma_i \gamma_j x_i \cdot x_j$$

Support vector machine can

estimate inner products in ϕ space

$$L = \sum_{i=0}^{m-1} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j \gamma_i \gamma_j n_i \cdot n_j$$

$\Downarrow$

$$L = \sum_{i=0}^{m-1} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j \gamma_i \gamma_j \phi(n_i) \cdot \phi(n_j)$$

Support vector machine can

estimate inner products in ϕ space

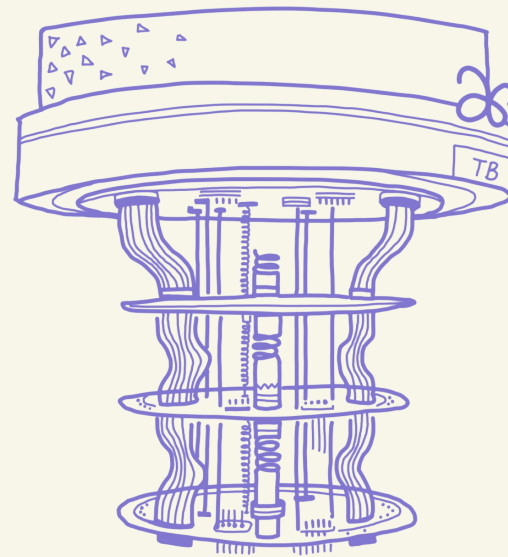
$$L = \sum_{i=0}^{m-1} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j \gamma_i \gamma_j x_i \cdot x_j$$

$\Downarrow$

$$L = \sum_{i=0}^{m-1} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j \gamma_i \gamma_j \phi(x_i) \cdot \phi(x_j)$$

$$L = \sum_{i=0}^{m-1} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j \gamma_i \gamma_j$$

$$\phi(n_i) \cdot \phi(n_j)$$



$$L = \sum_{i=0}^{m-1} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j \gamma_i \gamma_j$$

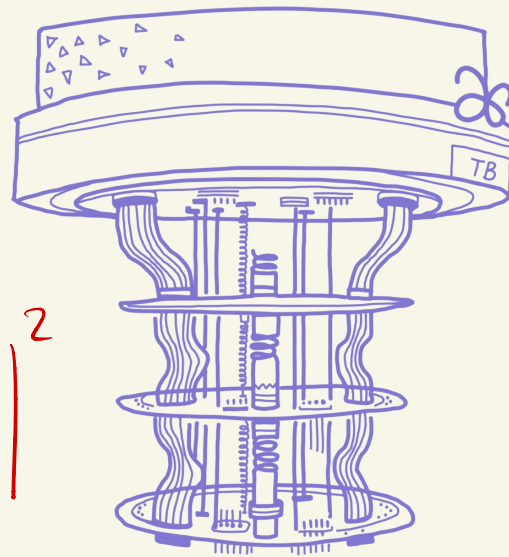
$$\phi(n_i) \cdot \phi(n_j)$$



How :

1) Map n_i to $|\phi(n_i)\rangle$

2) Estimate $|\langle \phi(n_i) | \phi(n_j) \rangle|^2$



classical data encoding



Quantum Feature map

$$x_i \xrightarrow{U} |\phi(x_i)\rangle$$

Imagine $x = (x_1, x_2, \dots, x_K)$

F features s.t. $x_i \in \mathbb{R}$

Easiest way of encoding x ?

Imagine $x = (x_1, x_2, \dots, x_K)$

$\forall$ features s.t. $x_i \in \mathbb{R}$

Translate x_i to binary (K bits)

$x \mapsto \left[\underbrace{010\dots0}_{x_1} \underbrace{101\dots1}_{x_2} \dots \right]$

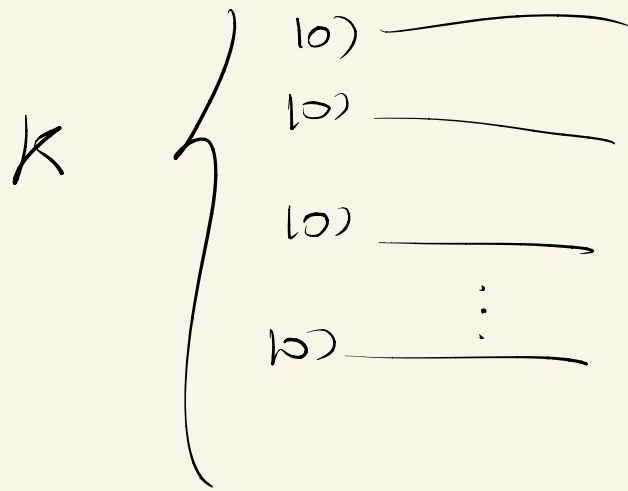
Imagine $x = (x_1, x_2, \dots, x_K)$

F features s.t. $x_i \in \mathbb{R}$

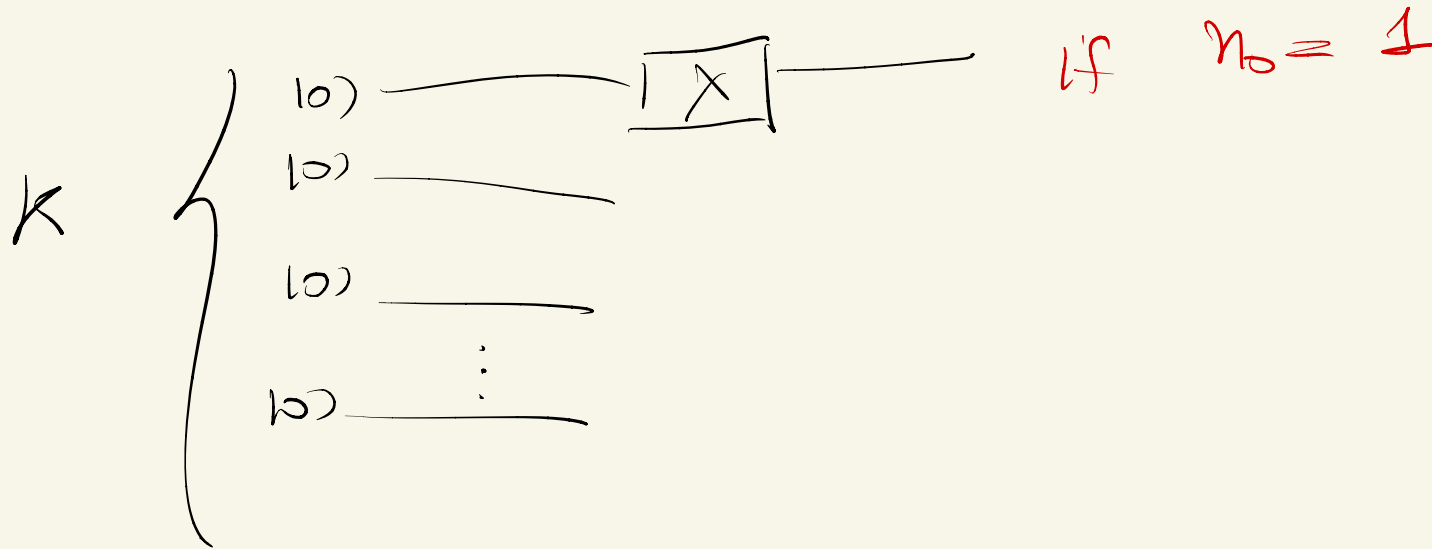
Translate x_i to binary (k bits)

$$|x\rangle = \left| \underbrace{010\dots0}_{x_1} \underbrace{101\dots1}_{x_2} \dots \right\rangle$$

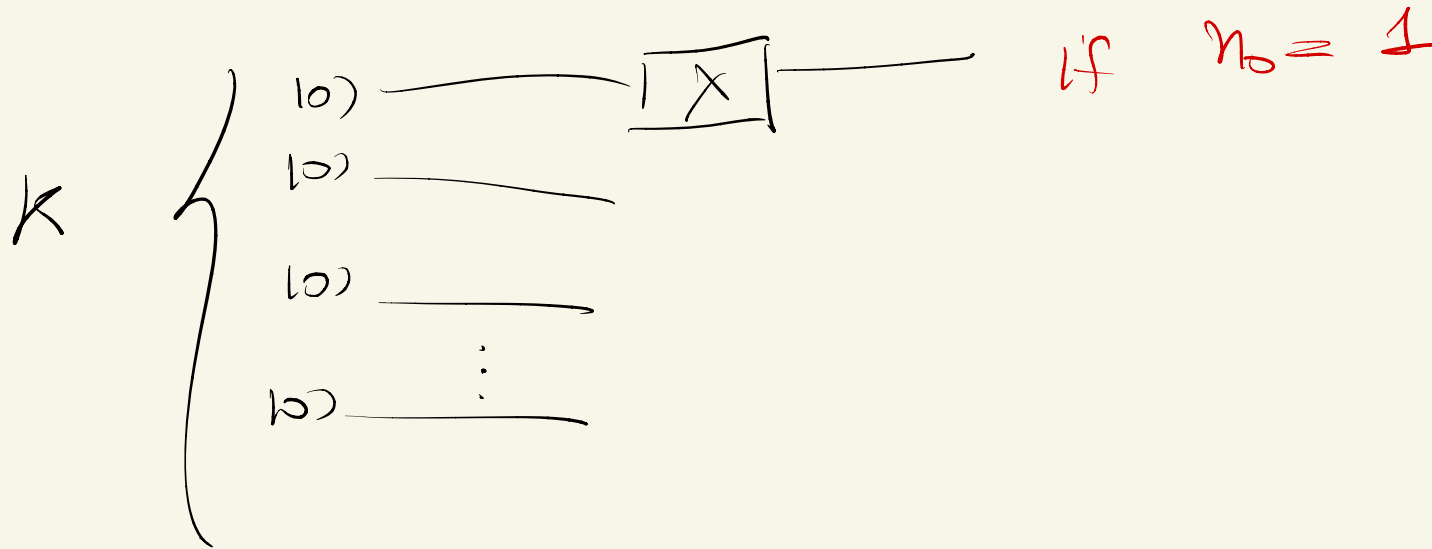
Binary encoding



Binary encoding



Binary encoding



qme.BasisEmbedding (binary, wires)

$O(FK)$ qubits

SAVE Qubits through superposition

instead converting to binary



Amplitude Encoding

Imagine $n = (n_1, n_2, \dots, n_K)$

F features s.t. $n_i \in \mathbb{R}$

F features $\longrightarrow$ $\log_2$ qubits s.t.

$$|n\rangle = \sum_{i=0}^{\log_2 F - 1} \tilde{n}_i |i\rangle$$

Imagine $n = (n_1, n_2, \dots, n_K)$

F features s.t. $n_i \in \mathbb{R}$

F features $\longrightarrow$ $\log \#$ qubits s.t.

$$|n\rangle \approx \sum_{i=0}^{\log F - 1} \tilde{n}_i |i\rangle$$

Imagine $n = (n_1, n_2, \dots, n_K)$

F features s.t. $n_i \in \mathbb{R}$

F features $\longrightarrow$ $\log F$ qubits s.t.

$$|n\rangle = \sum_{i=0}^{\log F - 1} \tilde{n}_i |i\rangle$$

Normalized s.t. $\sum_i |\tilde{n}_i|^2 = 1$

Encoding is not trivial

In general $O(2^n)$ gates necessary

(See Mottone's Algorithm)

Encoding is not trivial

In general $O(2^n)$ gates necessary

(See Mottone's Algorithm)

Im1. Amplitude Embedding (features , wires)

Encoding is not trivial

In general $O(2^n)$ gates necessary

(See Mottone's Algorithm)

Im1. Amplitude Embedding (features , wires , pad with)

constant like 0 if $|f|$ not power of 2

ex: $n = (n_1, n_2, n_3, n_4, n_5)$

$|F| = 5 \rightarrow \text{need 3 qubits}$

ex: $n = (n_1, n_2, n_3, n_4, n_5)$

$|F| = 5 \rightarrow$ need 3 qubits

↓

but with 3 qubits
there are $2^3 = 8$ states

ex: $n = (n_1, n_2, n_3, n_4, n_5)$

$|F| = 5 \rightarrow$ need 3 qubits



but with 3 qubits
there are $2^3 = 8$ states



pad with 0

$|n\rangle = (\tilde{n}_1, \tilde{n}_2, \tilde{n}_3, \tilde{n}_4, \tilde{n}_5, 0, 0, 0)$

So far:



Basis Embedding $\rightarrow$ depth 1
 $\rightarrow O(|F|k)$ qubits

Amplitude Embedding $\rightarrow$ exp. # gates
 $\rightarrow O(\log |F|)$ qubits

So far:



Basis Embedding $\rightarrow$ depth 1
 $\rightarrow O(|F|k)$ qubits

Amplitude Embedding $\rightarrow$ Exp. # gates
 $\rightarrow O(\log |F|)$ qubits

Both of both worlds? Angle Embedding

Imagine $n = (n_1, n_2, \dots, n_K)$

F features s.t. $n_i \in \mathbb{R}$

Rotation gates:

$$R_y(\theta) = \begin{pmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix}$$

$\{x, y, z\}$

Imagine $n = (n_1, n_2, \dots, n_K)$

F features s.t. $n_i \in \mathbb{R}$

Rotation gates:

$\boxed{R_y(\theta)}$ $\begin{pmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix}$

$\boxed{R_y(\tilde{n}_i)}$

$\tilde{n}_i \in [0, 2\pi]$

Angle Embedding

$$\begin{array}{lcl} \text{---} & R_\delta(\tilde{n}_0) & \text{---} \\ \text{---} & R_\delta(\tilde{n}_1) & \text{---} \\ \text{---} & R_\delta(\tilde{n}_2) & \text{---} \\ & \vdots & \\ \text{---} & R_\delta(\tilde{n}_f) & \text{---} \end{array}$$

$$\delta \in \{n, \gamma, z\}$$

Angle Embedding

— $R_\phi(\tilde{n}_0)$ —
— $R_\phi(\tilde{n}_1)$ —
— $R_\phi(\tilde{n}_2)$ —
⋮
— $R_\phi(\tilde{n}_f)$ —

$O(|F|)$ gates

$O(|F|)$ qubits

qml AngleEmbedding (Features, wires)

$$\phi \in \{n, y, z\}$$

Angle Embedding

— $R_\phi(\tilde{n}_0)$ —
— $R_\phi(\tilde{n}_1)$ —
— $R_\phi(\tilde{n}_2)$ —
⋮
— $R_\phi(\tilde{n}_f)$ —

$O(|F|)$ gates

$O(|F|)$ qubits

qml.AngleEmbedding (Features, wires, rotation = 'Y')

rotation $\in \{ 'X', 'Y', 'Z' \}$

Quantum kernel

$$k(n, n') = |\langle \phi(n) | \phi(n') \rangle|^2$$

//

overlap between two
quantum states

$$k(n, n') = \begin{cases} 0, & \text{orthogonal} \\ 1, & \text{equal} \end{cases}$$

Quantum Kernel

$$k(n, n') = | \langle \phi(n) | \phi(n') \rangle |^2$$

//

overlap between two
quantum states

$$k(n, n') = \begin{cases} 0, & \text{orthogonal} \\ 1, & \text{equal} \end{cases}$$

Decompose $k(n, n')$ in quantum operation?

$$k(n, n') = |\langle \phi(n) | \phi(n') \rangle|^2$$

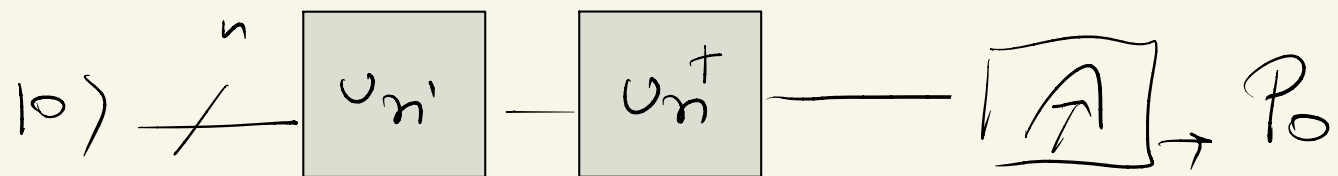
$$= |\langle 0 | \underbrace{U_n^\dagger} \cdot \underbrace{U_{n'}} | 0 \rangle|^2$$

$$= |\langle 0 | \psi(n, n') \rangle|^2$$

$$|\psi(n, n')\rangle = U_n^\dagger \cdot U_{n'} | 0 \rangle$$



Probability of the all-zero state



$U_n \equiv \left\{ \begin{array}{l} \text{Basis Embedding} \\ \text{Amplitude Embedding} \\ \text{Angle Embedding} \\ \vdots \end{array} \right.$