Supervised Learning with Quantum Computers

Isidora Araya Day
Radoica Draškić
Brennan Undseth

Delft University of Technology, The Netherlands

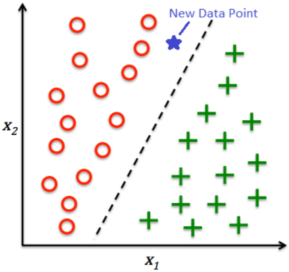
April 23, 2020

Overview

Table of Contents

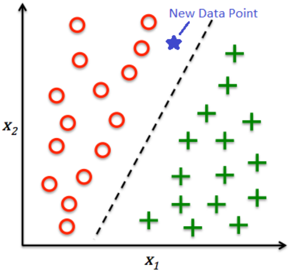
What is Machine Learning?

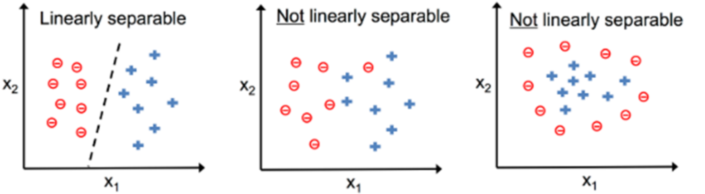
- Supervised learning
  - Input labeled data
  - Train machine
  - Label new data
- Unsupervised learning
  - Analysis of unlabeled data
- Reinforcement learning



What is Machine Learning?

- Supervised learning
  - Input labeled data
  - Train machine
  - Label new data
- Unsupervised learning
  - Analysis of unlabeled data
- Reinforcement learning





Supervised learning in breast cancer diagnosis

Discrimination between benign and malignant samples after fine needle biopsies [1].

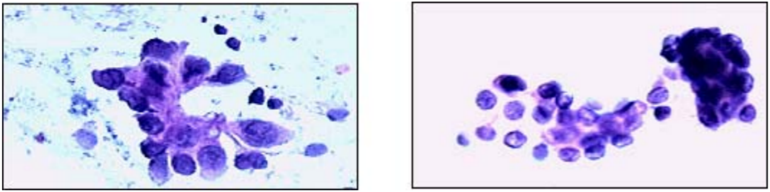


Figure: left:malignant, right: benign.

10 Parameters: radius, perimeter, area, compactness, smoothness, concavity, concave points, fractal dimension, texture of tumours.

[1] M. Sewak, P. Vaidya, C. Chan, and Zhong-Hui Duan (2007). "SVM Approach to Breast Cancer Classification". In: pp. 32–37

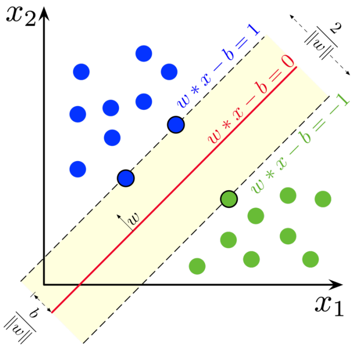
Support vector machine (SVM) [2]

Labeled data:

$$\{(\vec{x}_j, y_j) : \vec{x}_j \in \mathbb{R}^N, y_j = \pm 1\}_{j=1,\dots,M} \tag{1}$$

Purpose: to find a dividing hyperplane

$$y_j(\vec{w} \cdot \vec{x}_j + b) \geq 1 \tag{2}$$



[2] L. Saitta (1995). "Support-Vector Networks". In: 297, pp. 273–297

Table of Contents

Quantum machine learning

Motivation: Internal product and inverse matrix calculation [3].

In theory: HHL algorithm to solve linear systems of eq [4].

Bottleneck: Encoding.

In practice: Hybrid algorithms

A feature map that is **hard to estimate classically** is an important part of creating a quantum advantage

[3] P. Rebentrost, M. Mohseni, and S. Lloyd (2014). "Quantum Support Vector Machine for Big Data Classification". In: 130503.September, pp. 1–5. DOI: 10.1103/PhysRevLett.113.130503

[4] A. W. Harrow, A. Hassidim, and S. Lloyd (2009). "Quantum Algorithm for Linear Systems of Equations". In: 150502.October, pp. 1–4. DOI: 10.1103/PhysRevLett.103.150502

IBM - Hybrid approach [5]

2 different methods:

- Quantum Kernel Estimator
- Quantum Variational Classifier

[5] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta (2019). "Supervised learning with quantum-enhanced feature spaces". In: *Nature* 567.7747, 209–212. ISSN: 1476-4687. DOI: 10.1038/s41586-019-0980-2

IBM - Hybrid approach [5]

2 different methods:

- Quantum Kernel Estimator
- Quantum Variational Classifier

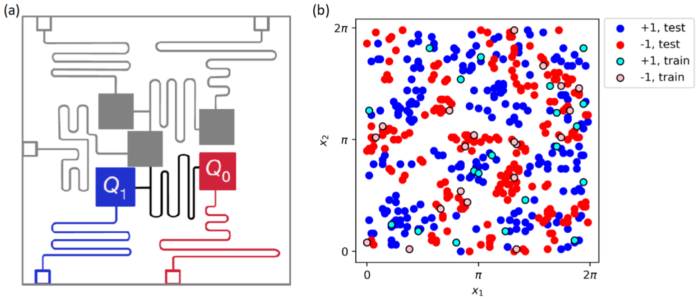
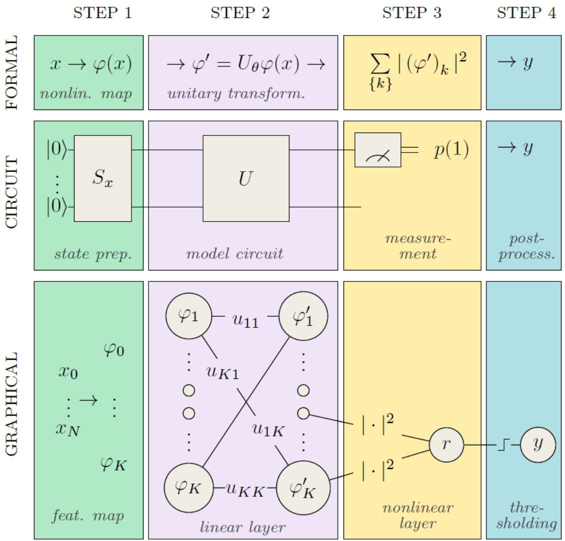


Figure: (a) 5 Qubit transmon circuit. (b) Results QKE Qiskit simulation.

[5] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta (2019). "Supervised learning with quantum-enhanced feature spaces". In: *Nature* 567.7747, 209–212. ISSN: 1476-4687. DOI: 10.1038/s41586-019-0980-2

Table of Contents

Workflow[6]



[6] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe (2020). "Circuit-centric quantum classifiers". In: *Phys. Rev. A* 101 (3), p. 032308. DOI: 10.1103/PhysRevA.101.032308

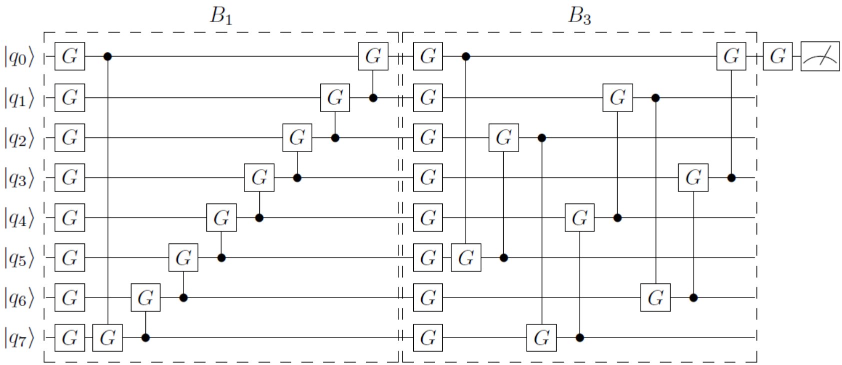
State preparation

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} \rightarrow \frac{1}{\sqrt{\sum_j x_j^2 + \sum_k |c_k|^2}} \begin{pmatrix} x_1 \\ \vdots \\ x_N \\ c_1 \\ \vdots \\ c_D \end{pmatrix} \equiv |\varphi(x)\rangle$$

$$|\varphi(x)\rangle \rightarrow \underbrace{|\varphi(x)\rangle \otimes \cdots \otimes |\varphi(x)\rangle}_{d \text{ times}}$$

Tensorial feature map will hopefully introduce nonlinearities that may facilitate the classification procedure.

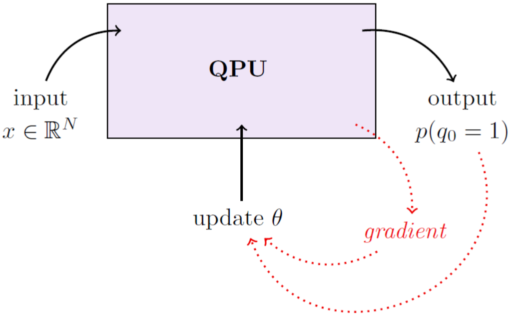
Variational circuit architecture



17 single qubit gates $G(\alpha, \beta, \gamma)$, 16 two qubit gates $CG(\alpha, \beta, \gamma)$.
Total number of parameters $33 \times 3 + 1 = 100$.

Training the model

- Prediction $\pi(x;\theta,b) = \langle\varphi(x)|U^\dagger(\theta)\,(Z\otimes\cdots\otimes I)\,U(\theta)|\varphi(x)\rangle + b$
- Cost function $\mathcal{C}(\theta,b) = \frac{1}{2M}\sum_{m=1}^{M}|\pi(x^m;\theta,b) - y^m|^2$
- Gradient descent $\theta_i^{(t)} = \theta_i^{(t-1)} - \eta\partial\mathcal{C}/\partial\theta_i$, $b^{(t)} = b^{(t-1)} - \eta\partial\mathcal{C}/\partial b$

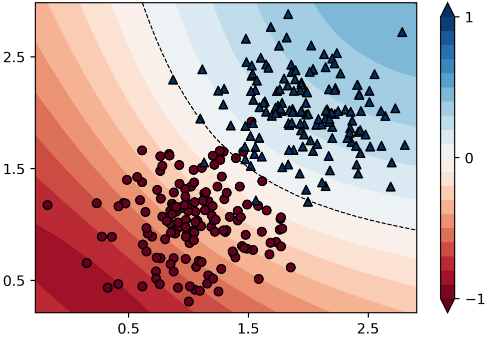


PENNYLANE

Qiskit

Linearly separable data

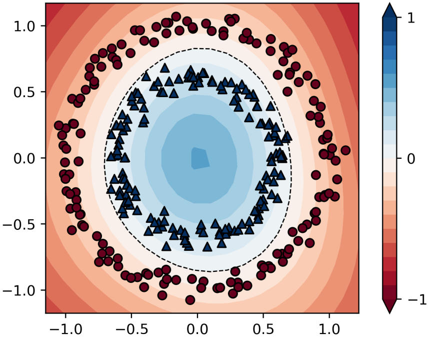
- 2 qubits
- padding with 2 uninformative features
- 97% accuracy



$$\begin{pmatrix} x & y \end{pmatrix}^T \to \begin{pmatrix} x & y & 1 & 1 \end{pmatrix}^T$$

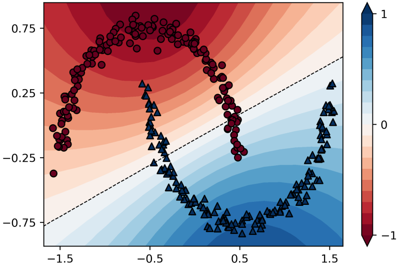
Concentric circles

- 4 qubits
- padding with 2 uninformative features
- $d = 2$
- 99% accuracy

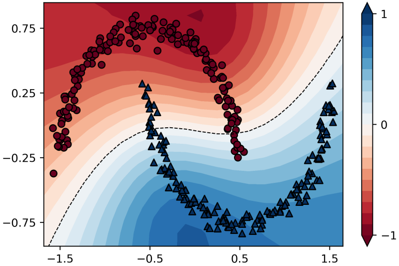


$$
\begin{aligned}
\begin{pmatrix} x & y \end{pmatrix}^T &\to \begin{pmatrix} x & y & 1 & 1 \end{pmatrix}^T \otimes \begin{pmatrix} x & y & 1 & 1 \end{pmatrix}^T \\
&= \begin{pmatrix} x^2 & xy & x & x & xy & y^2 & y & y & \dots \end{pmatrix}^T
\end{aligned}
$$

Moons



2 qubits
$d = 1$
80% accuracy



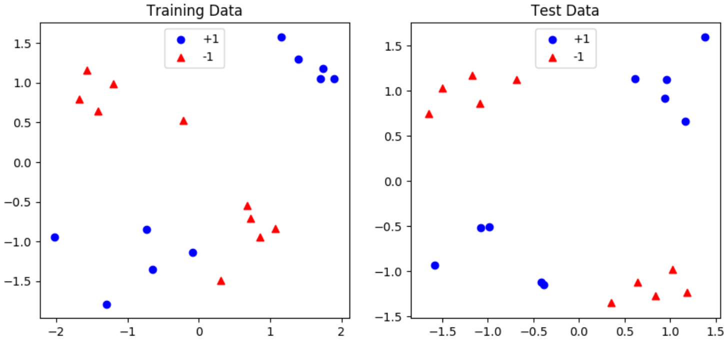
4 qubits
$d = 2$
96% accuracy

$$\begin{pmatrix} x & y \end{pmatrix}^T \to \begin{pmatrix} x & y & 1 & 1 \end{pmatrix}^T$$

$$\begin{pmatrix} x & y \end{pmatrix}^T \to \begin{pmatrix} x & y & 1 & 1 \end{pmatrix}^{\otimes 2, T}$$

Table of Contents

Exclusive-OR Data



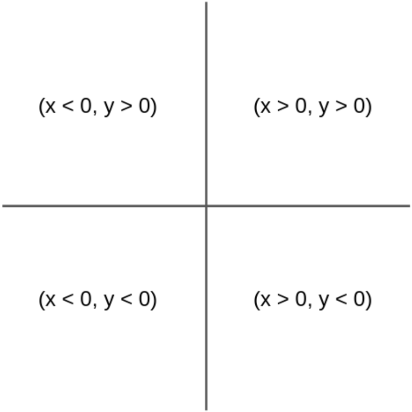
We use the training data to optimize the variational parameters $\vec{\theta}$ with respect to a **cost** function.
We use the test data to quantify the **accuracy** of our model.

State Preparation S_x

- Input in Cartesian coordinates: (x,y)
- Encode state in ket:

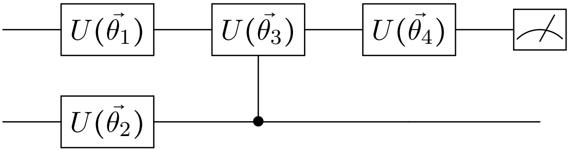
$$\frac{1}{\chi}\begin{pmatrix} x \\ y \end{pmatrix} \otimes \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{\chi}\begin{pmatrix} x^2 \\ xy \\ yx \\ y^2 \end{pmatrix}$$

- Note that states in quadrants II and IV pick up relative phase

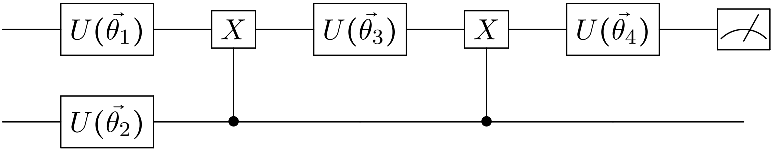


Variational Circuit $U(\vec{\theta})$

Since IBM Quantum Experience does not support arbitrary controlled unitaries...



We expand to the following circuit:



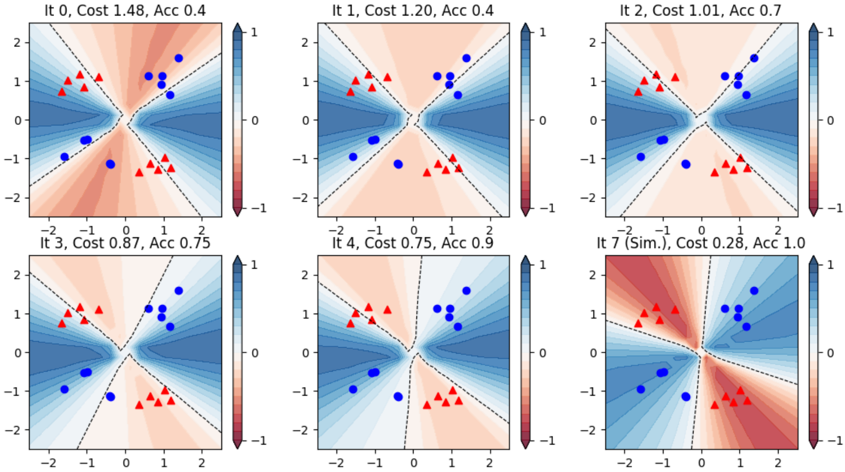
Model

- **Input** datum x^m with label y^m
- **Prediction** $\pi^m = \langle 0|S_x^\dagger U^\dagger(\vec{\theta}) Z_1 U(\vec{\theta}) S_x|0\rangle + b$
- **Optimize** $\vec{\theta}, b$ to minimize the **cost** function: $C = \frac{1}{2M}\sum_i^M (\pi^m - y^m)^2$
- State preparation and optimization is handled easily with Pennylane

PENNYLANE

```python
@qml.qnode(dev)
def circuit(weights, x=None):
    statepreparation(x)

    qml.Rot(weights[0, 0], weights[0, 1], weights[0, 2], wires=0)
    qml.Rot(weights[1, 0], weights[1, 1], weights[1, 2], wires=1)
    qml.CNOT(wires=[1,0])
    qml.Rot(weights[2, 0], weights[2, 1], weights[2, 2], wires=0)
    qml.CNOT(wires=[1,0])
    qml.Rot(weights[3, 0], weights[3, 1], weights[3, 2], wires=0)

    return qml.expval(qml.PauliZ(0))

def model(var, x=None):
    weights, bias = var
    return circuit(weights, x=x) + bias

def loss(labels, predictions):
    loss = 0
    for l, p in zip(labels, predictions):
        loss = loss + (l - p) ** 2
    loss = loss / len(labels)
    return loss
```

Results with IBM Quantum Experience



Notes on Implementation

IBM Q™

- IBM queue takes anywhere between seconds to hours
- Single iteration: 20 data points with 12 parameters requires 480 unique circuits
- Each iteration takes well over 1 day with 5 minute queue time
- Python script run on an Amazon cloud computer for a week

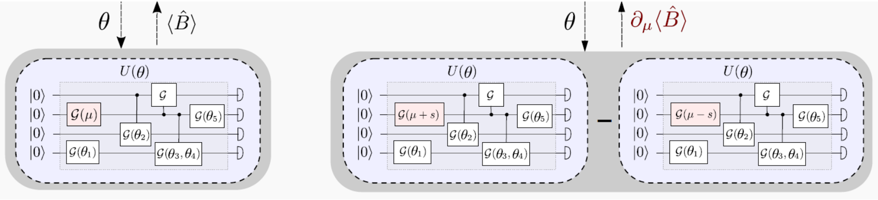
Conclusion

- Quantum computers can perform supervised machine learning tasks with existing quantum hardware, but dedicated computers will be required for meaningful tasks
- State encoding and optimization are non-trivial design choices and greatly influence classifier performance
- Many approaches and opportunities for quantum speedup, we have presented just one way

Thank you!
Questions?

Calculating the gradient

Shift rule



If f is some linear function of the expectation value $\langle \hat{B} \rangle$, and $G(\alpha, \beta, \gamma)$

$$\frac{\partial f(G(\mu))}{\partial \mu} = \frac{1}{2} \left[f(G(\mu + \pi/2)) - f(G(\mu - \pi/2)) \right], \; \mu \in \{\alpha, \beta, \gamma\}.$$

Same result holds for $CG(\alpha, \beta, \gamma)$.

IBM - extra

Kernel matrix:

$$K(\vec{x}, \vec{z}) = |\langle\Phi(\vec{x})|\Phi(\vec{z})\rangle|^2$$

$$U_{\Phi(x)} = \exp\left(i \sum_{S \subseteq [2]} \phi_S(x) \Pi_{i \in S} Z_i\right)$$

$$\phi_{\{i\}} = x_i \text{ and } \phi_{\{1,2\}} = (\pi - x_1)(\pi - x_2)$$