Intro to Quantum ML

CS4/591 HPC - Fall 2025

Overview

What *is* machine learning?

What *is* quantum computing?

“When mixing machine learning with ‘quantum,’ you catalyse a hype-condensate.”

- Jacob Biamonte, prominent contributor to QML Research

There is a lot of buzz and **skepticism** around quantum ML, but what is it?

What is Machine Learning?

So what’s the fuss?

- QML may provide the edge over classical ML given future tech innovation…
  - … provided sufficient improvement of quantum hardware.
- QML takes advantage of methods hard to simulate classically, e.g. superposition.
- Insufficient hardware and attention to date, make these methods less understood.
- Recent innovations in QML suggest drastic improvement within years.
- `qiskit` has built-in methods making QML accessible to new researchers.

Support Vector Classifiers (SVC)

- Classical supervised learning method for classification and regression.
- Transforms data to a higher dimension for improved separability.
- Trains a hyperplane in higher dimension to classify data points.
- Utilizes the kernel trick to work in the original feature space.
- *Versatile* by the creativity of the applied kernel.
- The trained partitioning hyperplane is the line with the lowest cost that is farthest from the closest data points of each class.
- Can choose if the cost of one incorrectly labelled class outweighs another.

SVC - Projection

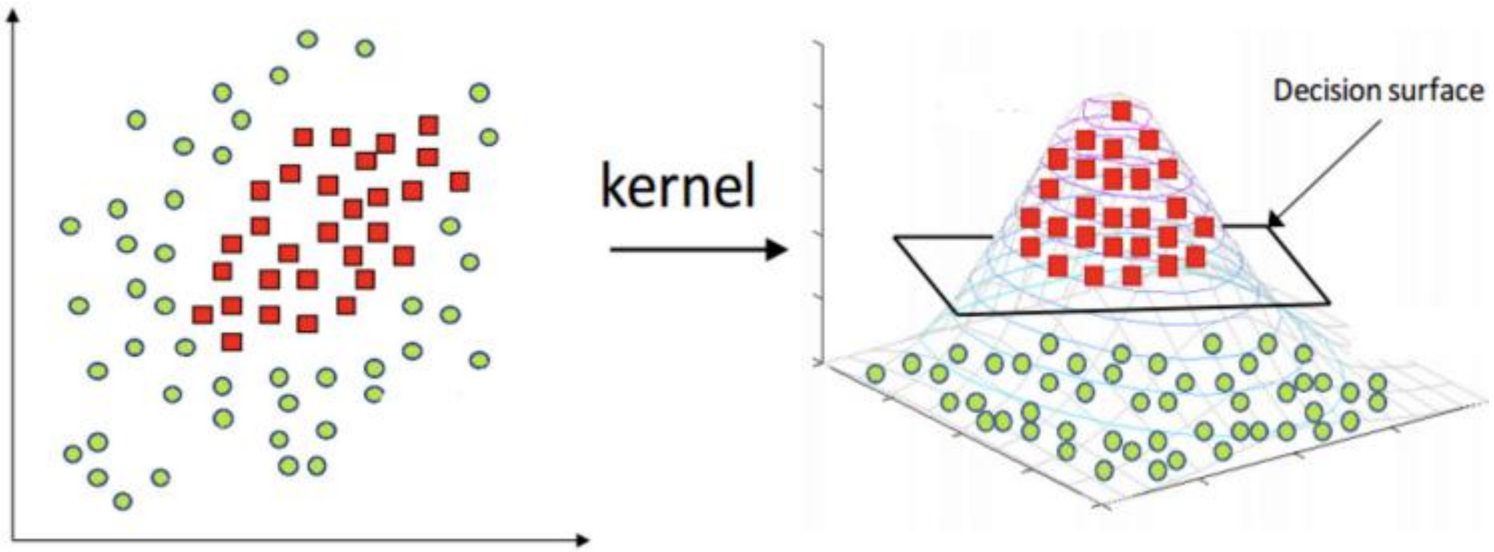

Fig 1: Hyperplane in projected space

SVC – Different Kernels with Iris Dataset

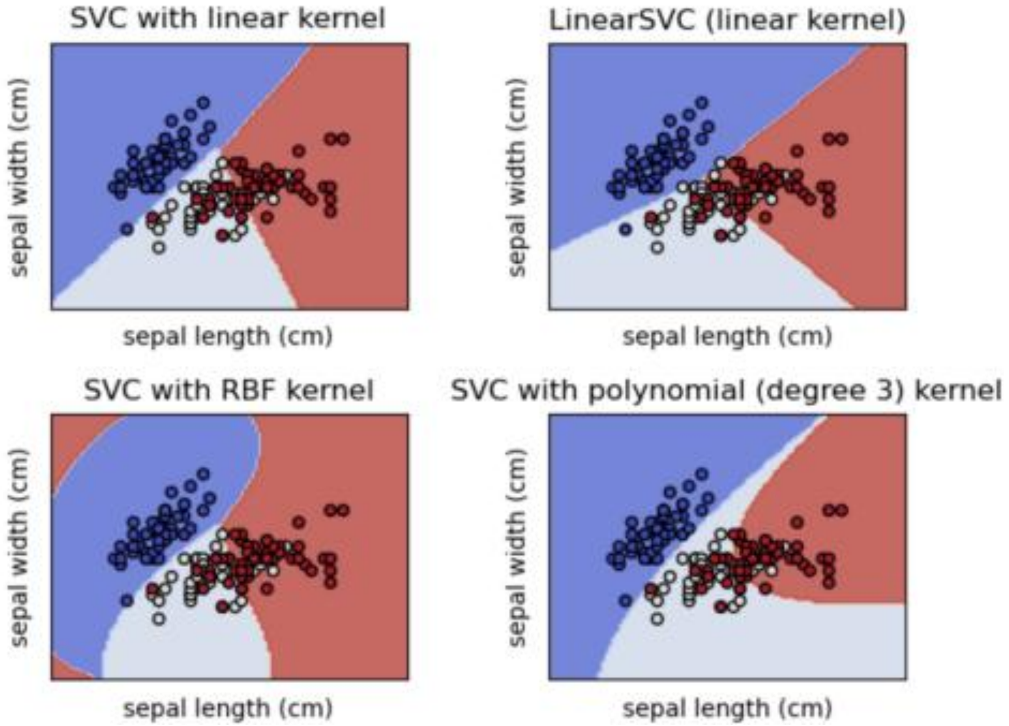

Fig 2: SVC decision boundaries with different kernels. Source: Scikit-learn

Kernel

- A Kernel function can be represented by a matrix.
- Classical kernel matrix: $K = [\text{all } K_{ij}] = [\text{all } \langle f(\vec{x}_i), f(\vec{x}_j)\rangle]$; f is map.
- Quantum kernel matrix: $K = [\text{all } K_{ij}] = [\text{all } |\langle \phi(\vec{x}_i)|\phi(\vec{x}_j)\rangle|^2]$; ϕ is feature map. → This is just the fidelity between (pure) states!
- `Qiskit` has 2 kernel interfaces:
  - BaseKernel: implemented as `FidelityQuantumKernel`.
  - TrainableKernel: implemented as `TrainableFidelityQuantumKernel`.
- A trainable kernel allows training of parameters of the feature map.

QML Building Blocks

- Current QML algorithms typically consist of three blocks.
  - Quantum encoding of the data → feature map
  - A Quantum algorithm → kernel, QNN, …
  - Classical post processing → SVC, optimizer, …
- A classical optimizer iteratively optimizes the quantum steps.
- QML currently is *hybrid quantum classical* computing.

SVC versus QSVC

SVC	QSVC
Kernel map: Implicitly map data to higher dimensional space. → Can choose optimal kernel map for given data.	Feature map: Parametrized circuit maps data to higher dimensional quantum space. → Feature map might be better than any classical kernel.
Evaluate kernel matrix (similarities represented by dot products) → With kernel trick efficient on classical computer	Evaluate quantum kernel matrix (similarities represented by overlap / fidelity) → Efficient on quantum computer.
On a classical computer: Train the hyperplane or use it to classify new data.	

Quantum SVC

- The QSVC provided by `qiskit` is an extension of the `scikit-learn` SVC.
- QSVC with q-kernel gives same results as SVC with evaluated q-kernel.
- With current quantum tech, QSVC good enough, but not as great as SVC yet.
- Promises better results with more qubits, less noise, better quantum kernels.
- Quantum Kernel Alignment (QKA) provides further chance to improve QSVC:
    - Additionally optimizes a trainable feature map.
    - Can take existing feature map, add more parameterized gates, and likely produce at least an improved score.

Let's Try It Out!

- In your terminal (on **Easley** or **Hopper**):

```
git clone git@github.com:alexknigge/teaching.git
```

- Then, go to:

easley.alliance.unm.edu

- Start a Jupyter Notebook server for an hour
    - Navigate to the directory `teaching/qml`

Clustering

- Unsupervised learning method (i.e. unlabelled data).
- Use some metric to decide which points belong together in how many classes.
- Can also define the amount of classes to be split among.
- Many clustering methods: KMeans, Spectral, Mean Shift, Gaussian Mixtures, etc.
- qiskit does not have extension, but evaluated q-kernels can be applied.
- Examples of other quantum-adapted classifiers:
    - Quantum Decision Tree Classifiers, Quantum Nearest Neighbor Classifiers.

Classical vs Quantum

	Classical	Quantum
Advantage	Highly stable and powerful hardware	High parallelism through superposition and entanglement, exponential compute power
Limitations	Inferior compute memory at scale	Slow I/O, need to convert classical data to quantum, noise
SVMs	Choose optimal kernels to map data to higher dimensions	Use quantum feature maps with entanglement to map data to possibly even better representation

Exercises

- There is way more to do inside of the Jupyter Notebook provided
- Additionally, if you are further curious about this, read the original tutorial given at the Supercomputing Conference (SC24) here:

https://gitlab.gwdg.de/ag-compute-public/sc-2024-qml-tutorial

References

- Patrick Gelss, Christian Boehme, Lourens van Niekerk.

  https://gitlab.gwdg.de/ag-compute-public/sc-2024-qml-tutorial

- Sebastian Raschka, Yuxi (Hayden) Liu, and Vahid Mirjalili.

  Machine Learning with Pytorch and Scitkit-Learn

- Fireship, *Machine Learning Explained in 100 Seconds.*

  https://www.youtube.com/watch?v=PeMlggyqz0Y