Report Week 4

In week 3, I studied how Quantum Support Vector Machine is implemented through the paper Quantum Support Vector Machine. The paper showed that a quantum support vector machine can be implemented with $O(\log MN)$ run time in both training and classification stages. In week 3 report, I covered how to get the hyperparameters b and α to construct the hyperplane. In this week report, I will cover classification steps of QSVM.

QSVM Classification

Given a test example $\vec{x_o}$, SVM classifies $\vec{x}$ as:

$$y(\vec{x_o}) = sign(\vec{w} * \vec{x} + b)$$

Recall that when solving for KKT conditions we get the following result:

$$w = \sum_{i=1}^{M} \alpha_i \vec{x_i}$$

Hence,

$$y(\vec{x_o}) = sign(\sum_{i=1}^{M} \alpha_i (\vec{x_i} * \vec{x_o}) + b)$$

Let:

$$|0, \vec{y}\rangle = \frac{1}{\sqrt{N_{0,y}}}(|0\rangle + \sum_{i=1}^{M} y_i |i\rangle \text{, where } N_{0,y} = 1 + \sum_{i=1}^{M} |y_i|^2$$

Apply F^{-1} on this we obtain:

$$|b, \vec{\alpha}\rangle = \frac{1}{\sqrt{N_{b,\alpha}}}(b|0\rangle + \sum_{i=1}^{M} \alpha_i |i\rangle), \text{ where } N_{b,\alpha} = b^2 + \sum_{i=1}^{M} {\alpha_i}^2$$

Now, let's start constructing training data state and query state for classification. From the above state, construct the training data state:

$$|\widetilde{u}\rangle = \frac{1}{\sqrt{N_{\widetilde{u}}}}(b|0\rangle|0\rangle + \sum_{i=1}^{M} \alpha_i |\vec{x_i}| |k\rangle |\vec{x_i}\rangle)$$

with $N_{\widetilde{u}} = b^2 + \sum_{i=1}^{M} {\alpha_i}^2 |\vec{x_i}|^2$
In addition, we construct the query state:

$$|\widetilde{x}\rangle = \frac{1}{\sqrt{N_{\widetilde{x}}}}(|0\rangle|0\rangle + \sum_{i=1}^{M} |\vec{x}| |k\rangle |\vec{x}\rangle)$$

with $N_{\widetilde{x}} = M|\vec{x}|^2 + 1$. We have that:

$$\langle\widetilde{u}|\widetilde{x}\rangle = \frac{1}{\sqrt{N_{\widetilde{u}} N_{\widetilde{x}}}}(b + \sum_{i=1}^{M} \alpha_i |\vec{x_i}||\vec{x}| \langle\vec{x_i}|\vec{x}\rangle)$$

Hence, we would have that $y(\vec{x}) = sign(\langle\widetilde{u}|\widetilde{x}\rangle)$. If $\langle\widetilde{u}|\widetilde{x}\rangle > 0$, then $|\vec{x}\rangle$ would be classified as +1, otherwise, -1. The inner product, $\langle\widetilde{u}|\widetilde{x}\rangle$ can be computed in $O(\log NM)$ run time.

Swap Test

One of the most important immediate step in computing inner product as described in [2] involves the swap test:

$$|0\rangle|\psi\rangle|\varphi\rangle \xrightarrow{\text{C-SWAP}} |0\rangle|\psi\rangle|\varphi\rangle$$
$$|1\rangle|\psi\rangle|\varphi\rangle \xrightarrow{\text{C-SWAP}} |1\rangle|\varphi\rangle|\psi\rangle$$

Swap Test is most commonly used to test for equality of two unknown quantum states, i.e., estimates their inner product. To perform a swap test on $|\psi\rangle$ and $|\varphi\rangle$ means to do the following:

1. Initialize the state $|0\rangle|\psi\rangle|\varphi\rangle$
2. Apply the Hadamard gate to the first register
3. Apply C-SWAP to the three registers
4. Apply a Hadamard gate to the first register
5. Measure the first register
6. "Accept" if the outcome is $|0\rangle$ and "reject" if the outcome is $|1\rangle$

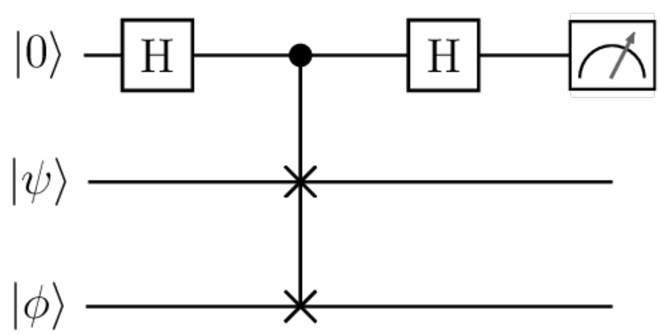

Figure 1: Quantum Circuit of a Swap Test

The evolution of the system is as follow:

$$\begin{aligned} |0,\varphi,\psi\rangle &\to \frac{1}{\sqrt{2}}(|0,\varphi,\psi\rangle + |1,\varphi,\psi\rangle) \to \frac{1}{\sqrt{2}}(|0,\varphi,\psi\rangle + |1,\psi,\varphi\rangle) \\ &\to \frac{1}{2}(|0,\varphi,\psi\rangle + |1,\varphi,\psi\rangle + |0,\psi,\varphi\rangle - |1,\psi,\varphi\rangle) \\ &= |0\rangle \otimes \frac{1}{2}(|\varphi,\psi\rangle + |\psi,\varphi\rangle) + |1\rangle \otimes \frac{1}{2}(|\varphi,\psi\rangle - |\psi,\varphi\rangle) \end{aligned} \tag{1}$$

Hence, the probability of observing $|0\rangle$ is:

$$
\begin{aligned}
Pr(\text{observing } |0\rangle) &= \frac{1}{4}(\langle\varphi,\psi| + \langle\psi,\varphi|)(|\varphi,\psi\rangle + |\psi,\varphi\rangle) \\
&= \frac{1}{4}(2 + \langle\psi,\varphi|\varphi,\psi\rangle + \langle\varphi,\psi|\psi,\varphi\rangle) \\
&= \frac{1}{2} + \frac{1}{2}|\langle\psi|\varphi\rangle|^2
\end{aligned} \tag{2}
$$

In the case of our classification problem, we can construct two states:

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle|\widetilde{u}\rangle + |1\rangle|\widetilde{x}\rangle)$$

$$|\phi\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

Performing a swap test on $|\psi\rangle$ and $|\phi\rangle$ reveals

$$|\langle\psi|\phi\rangle|^2 = 2 * Pr(\text{observing } |0\rangle) - 1$$

And we also have that

$$|\langle\psi|\phi\rangle|^2 = \frac{1}{2}(1 - \langle\widetilde{u}|\widetilde{x}\rangle)$$

Solving for Linear Equation

Classically, the problem of linear equations is to find an unknown vector $\vec{x}$ in the following linear equation: $A\vec{x} = \vec{b}$ for a known $N \times N$ matrix A, and a known N-dimensional vector $\vec{b}$. The quantum version of the problem can be written as $A|x\rangle = b$, where A is a Hermitian. If A is not Hermitian, one can solve

$$\begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} * \begin{bmatrix} 0 \\ \vec{x} \end{bmatrix} = \begin{bmatrix} \vec{b} \\ 0 \end{bmatrix}$$

instead of the original equation, so the algorithm is general for any invertible matrices.
The matrix A and the states $|x\rangle$ and $|b\rangle$ can be expanded in terms of the eigenstates of A as

$$A = \sum_{j=1}^{N} \lambda_j |u_j\rangle |u_j\rangle \quad A^{-1} = \sum_{j=1}^{N} {\lambda_j}^{-1} |u_j\rangle \langle u_j|$$

$$|b\rangle = \sum_{j=1}^{N} \beta_j |u_j\rangle \quad \text{where} \quad \beta_j = \langle u_j|b\rangle \quad \text{, and}$$

$$|x\rangle = A^{-1}|b\rangle = (\sum_{i=1}^{N} {\lambda_k}^{-1} |u_k\rangle \langle u_k|)(\sum_{j=1}^{N} \beta_j |u_j\rangle) = \sum_{j=1}^{N} \frac{\beta_j}{\lambda_j} |u_j\rangle$$

where λ_j and $|u_j\rangle$ are the eigenvalues and eigenstates of A. Generally, the quantum algorithms for solving the linear equation can be described as follows.

1. Prepare n register qubits in $|0\rangle^{\otimes n}$ state with the vector qubit $|b\rangle$:

$$|b\rangle |0\rangle^{\otimes n} = \sum_{j=1}^{N} \beta_j |u_j\rangle |0\rangle^{\otimes n}$$

2. Performing quantum phase estimation on the above state, i.e. apply $e^{-iAt} for varying times$, we obtain the following state $\sum_{j=1}^{N} \beta_j |u_j\rangle |\lambda_j\rangle$

3. Introduce an ancilla qubit $|0\rangle$, and perform the R_Y rotation on the ancilla, where $R_y = e^{\frac{-i\theta}{2Y}}$. The total rotation angle will be determined by the eigenvalue stored in the register qubit such that

$$\sum_{j=1}^{N} \beta_j |u_j\rangle |\lambda_j\rangle |0\rangle \rightarrow \sum_{j=1}^{N} \beta_j |u_j\rangle |\lambda_j\rangle (\sqrt{1 - \frac{C^2}{\lambda_j^2}} |0\rangle + \frac{C}{\lambda_j} |1\rangle)$$

4. Perform inverse phase estimation to disentangle the eigenvalue registers:

$$\sum_{j=1}^{N} \beta_j |u_j\rangle |\lambda_j\rangle (\sqrt{1 - \frac{C^2}{\lambda_j^2}} |0\rangle + \frac{C}{\lambda_j} |1\rangle)$$

5. Postselect for the ancillary qubit of $|1\rangle$:

$$\sum_{j=1}^{N} \beta_j |u_j\rangle \frac{C}{\lambda_j} |1\rangle$$

Others

I looked at the code from , but after Keita's presentation last Wednesday, I believed QISKIT is a better tool. I haven't had the chance to work on Qiskit until early this week. Right now, I'm working on the code for QSVM with QISKIT

Up Coming

Next week, I will finish the code for QSVM and start working on Quantum Neural Networks, and Quantum Deep Learning.

References

[1] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. *Quantum Support Vector Machine*

[2] Seth Lloyd, Massoud Mohseni, and Patrick Rebentrost. Quantum Algorithms for Supervised and Unsupervised Machine Learning