Milestone 2 Progress Report: Quantum Clustering Educational Module

May 30, 2025

Contents

1 Overview

Milestone 2 fixes the main problems from milestone 1 feedback. We made key improvements to make the module better for learning.

2 Key Improvements Made

2.1 Better Data

Changed from too-perfect data to messy synthetic data. This shows why we need clustering algorithms and gives students a more realistic learning experience.

2.2 Better Explanations

Added detailed explanations for quantum concepts that were missing in milestone 1:

- Analysis of quantum K-means implementation with angle encoding
- Clear explanation of fidelity and its mathematical formula
- Introduction to Swap Test for distance computation
- Explanation of why Swap Test is more sophisticated than direct fidelity measurement

2.3 Benchmarking: Classical vs Quantum Clustering

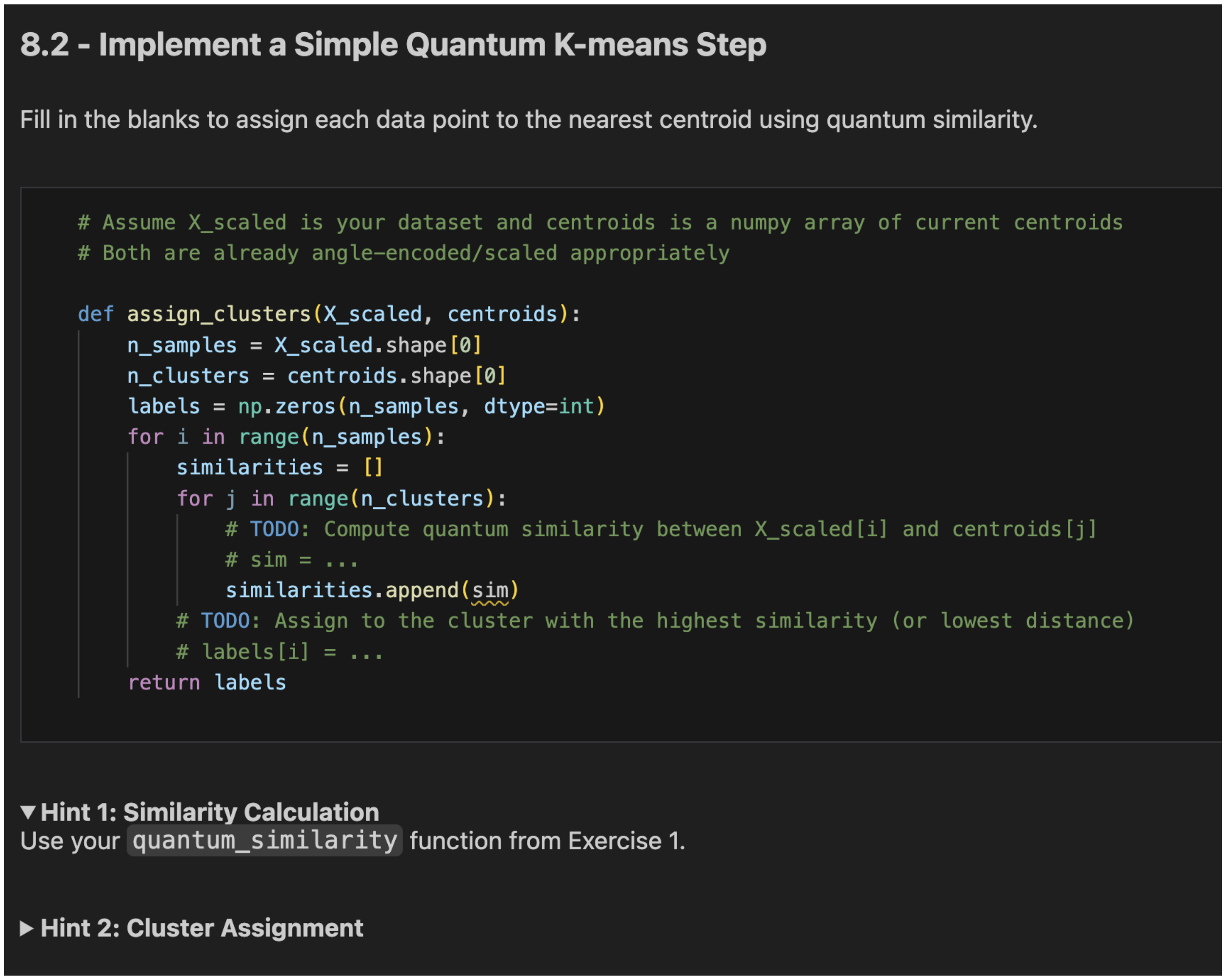
Added a section that benchmarks the performance of classical and quantum clustering algorithms.

- Benchmarking the performance of classical and quantum clustering algorithms
- Silhouette score comparison
- Inertia (SSE) comparison
- Runtime comparison

2.4 Complete Exercise Section

Built a full exercise section with:

- Code templates that students fill in
- Helpful hints when students get stuck
- Easy and hard exercises for different skill levels



8.2 - Implement a Simple Quantum K-means Step

Fill in the blanks to assign each data point to the nearest centroid using quantum similarity.

```
# Assume X_scaled is your dataset and centroids is a numpy array of current centroids
# Both are already angle-encoded/scaled appropriately

def assign_clusters(X_scaled, centroids):
    n_samples = X_scaled.shape[0]
    n_clusters = centroids.shape[0]
    labels = np.zeros(n_samples, dtype=int)
    for i in range(n_samples):
        similarities = []
        for j in range(n_clusters):
            # TODO: Compute quantum similarity between X_scaled[i] and centroids[j]
            # sim = ...
            similarities.append(sim)
        # TODO: Assign to the cluster with the highest similarity (or lowest distance)
        # labels[i] = ...
    return labels
```

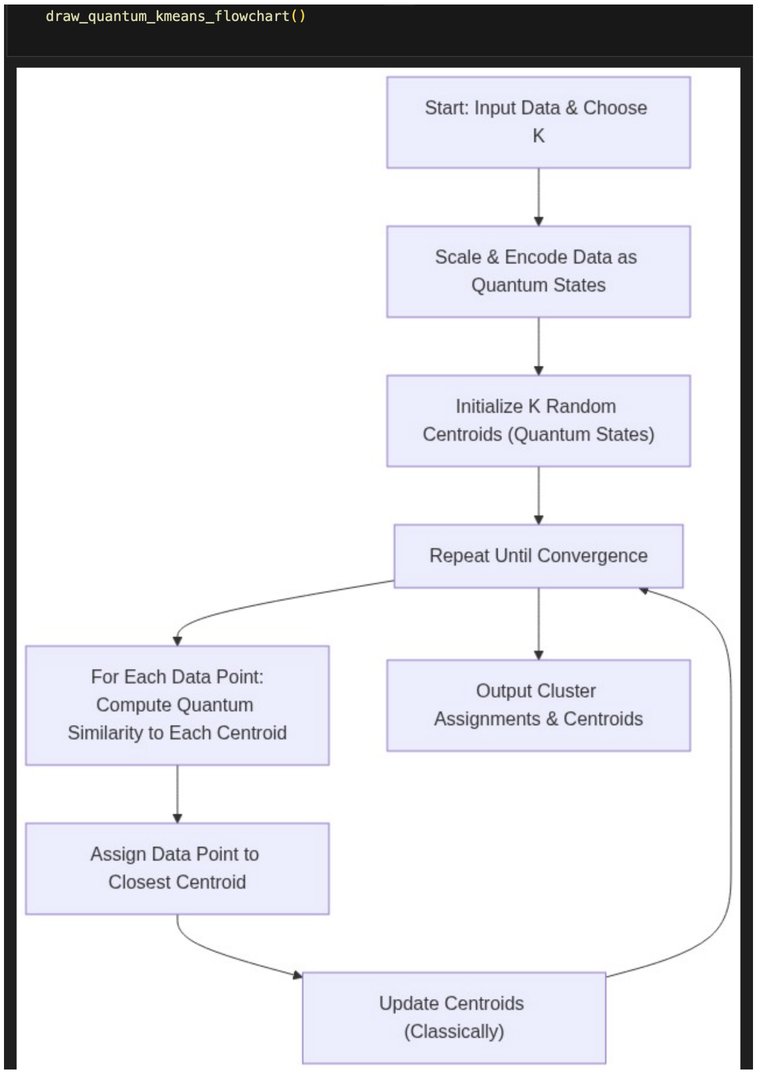
▼ **Hint 1: Similarity Calculation**
Use your `quantum_similarity` function from Exercise 1.

▶ **Hint 2: Cluster Assignment**

Figure 1: Exercise section in jupyter notebook

2.5 Algorithm Flow Chart

Made a visual chart that shows how the quantum clustering algorithm works step by step. This helps students see the process clearly.



```
draw_quantum_kmeans_flowchart()
```

Start: Input Data & Choose K
Scale & Encode Data as Quantum States
Initialize K Random Centroids (Quantum States)
Repeat Until Convergence
For Each Data Point: Compute Quantum Similarity to Each Centroid
Output Cluster Assignments & Centroids
Assign Data Point to Closest Centroid
Update Centroids (Classically)

Figure 2: Algorithm flow chart

2.6 Introduction to Unsupervised Learning

Added a simple explanation of unsupervised learning at the beginning. This gives students the basic knowledge they need before learning about clustering.

3 Conclusion

These improvements fix the problems from milestone 1 feedback. The better data, clearer explanations, hands-on exercises, visual guide, and basic introduction work together to help students learn both the theory and practice of quantum clustering.