The Workings of Quantum Anomaly Detection (One-class Classifiers)

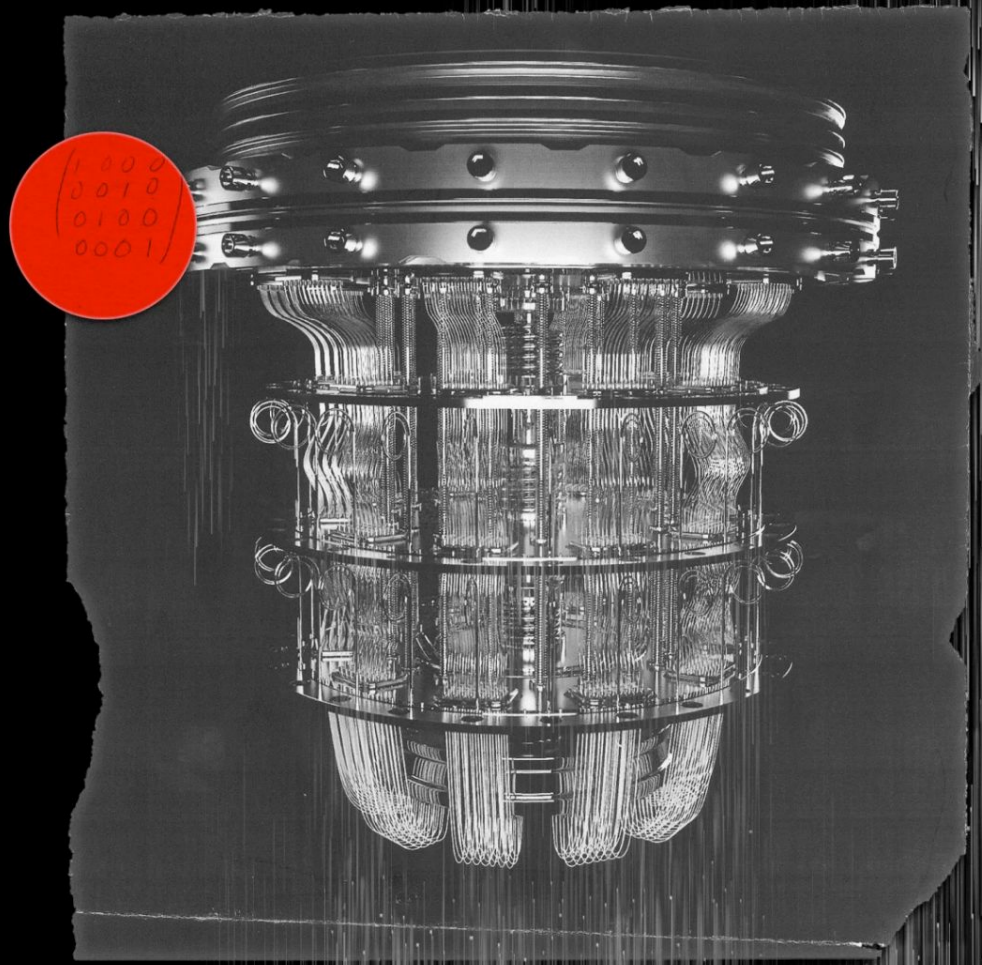
Final Presentation
30th June 2023
Group 21

Isabel Maria Adrover Cabot
Christos Koromilas
Ismael Gómez Garrido,
Gokul Jayakumar
Ivana Pallister

Maastricht University

Agenda

- Introduction
- Research Questions
- Implementation
  - Datasets
  - Classical models implementation
    - SVM
    - Kernel PCA
    - Autoencoder
  - Quantum Model implementation
    - QSVM
    - VQC
- Answer Research Questions
- Conclusion and future work



Quantum Computing to be worth 4.85 Billion USD by 2030

How does it compare to classical methods?

https://finance.yahoo.com/news/quantum-computing-market-reach-usd-121600293.html

Research Questions

1. How do **different datasets affect** the performance of a **quantum one class classification method**?

2. How does the **quantum approach** perform in **comparison** to **classical methods** depending on the **number of parameters**?

3. How do **quantum one-class classification methods** compare to c**lassical one-class classification methods** in terms of **performance** and **computational complexity**?

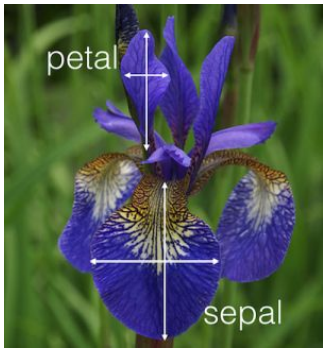
4. What **benefits** are apparent by using **quantum computing with one classification**?

Datasets

Progress: Datasets Research

Iris dataset

a. Dimensionality = 150 observations and 4 features
b. Classes = 3
c. Pre-processing:
    i. Select class 1 as inlier
    ii. Select [0,2] as outliers

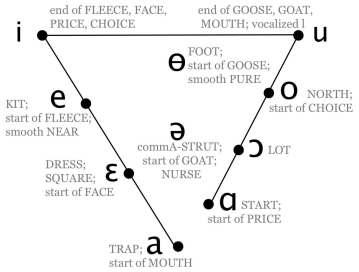


https://docs.kedro.org/en/0.18.1/get_started/example_project.html

Progress: Datasets Research

Vowels dataset

a. Dimensionality = 987 observations and 14 features
b. Classes = 11
c. Pre-processing:
   i. Select vowel 1 as inlier
   ii. Select [2,3,5,6,9] as outliers

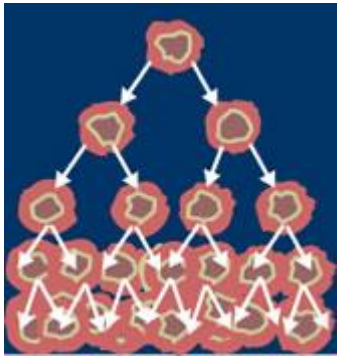


https://www.englishspeechservices.com/blog/british-vowels/

Progress: Datasets Research

Breast Cancer dataset

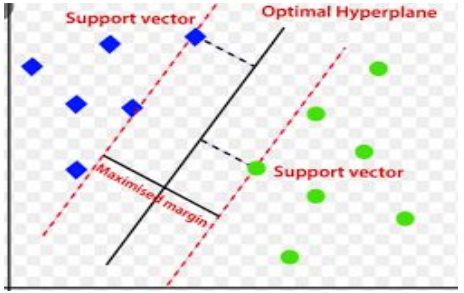
a. Dimensionality = 569 observations and 30 features
b. Classes 2
    i. We consider “benign” as inlier, “malign” as outlier



https://archive.ics.uci.edu/ml/datasets/breast+cancer

Classical Implementation

OCCSVM

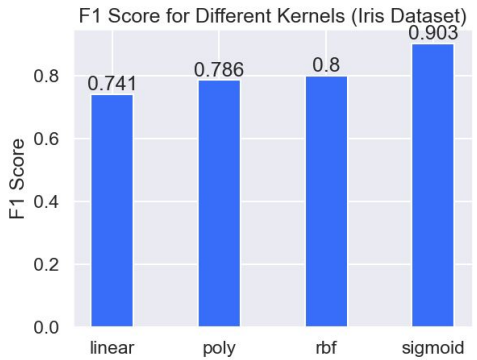


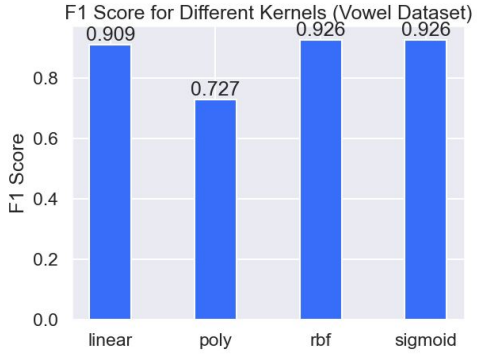
GridSearchCV with a cross-validation = 5

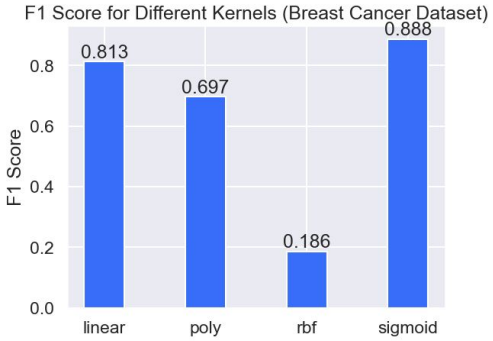
- **Kernel** : linear, poly, rbf, and sigmoid
- **Nu**: represents an **upper bound** on the fraction of training errors and a lower bound on the fraction of support vectors
- **Gamma**:which represents the kernel coefficient for ’rbf’, ’poly’, and ’sigmoid’ kernels

OCCSVM

Datasets	Iris	Vowels	Breast Cancer
Kernel	sigmoid	sigmoid	sigmoid
Gamma	0.5	0.5	1
Nu	0.2	0.2	0.1
Accuracy	0.875	0.897	0.897



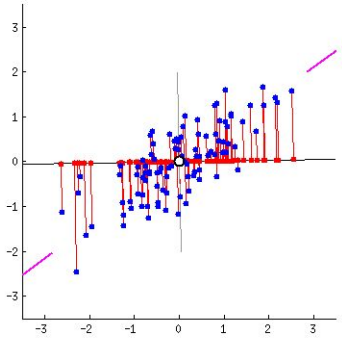




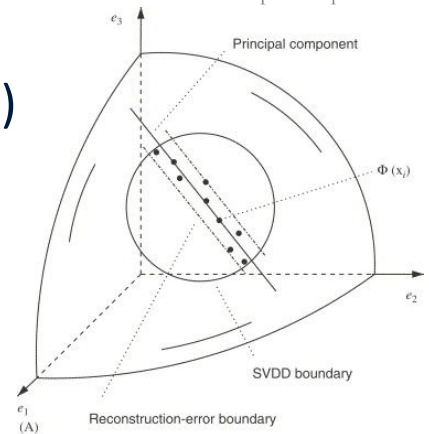
PCA and Kernel PCA

1. Progress implementing Classical: PCA and Kernel PCA
   a. Unsupervised AD: Outlier Detection (PyOD)
   b. Semi-Supervised AD: Novelty Detection

2. PCA and KernelPCA:
   a. Fit model with train-set: N or NA (~66%)
   b. Reconstruction Error
   c. Select Threshold
   d. Calculate Outlier/Anomaly Scores
   e. Classify test-set NA (~33%)



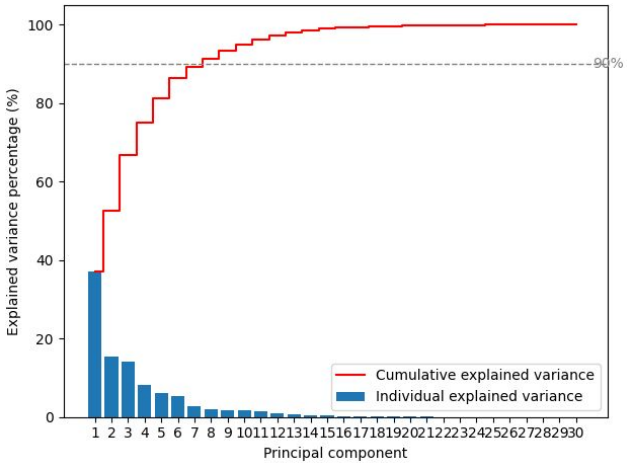
Projection of data points onto Principal Components. Source: Stack Exchange



https://doi.org/10.1016/j.patcog.2006.07.009

PCA and Kernel PCA

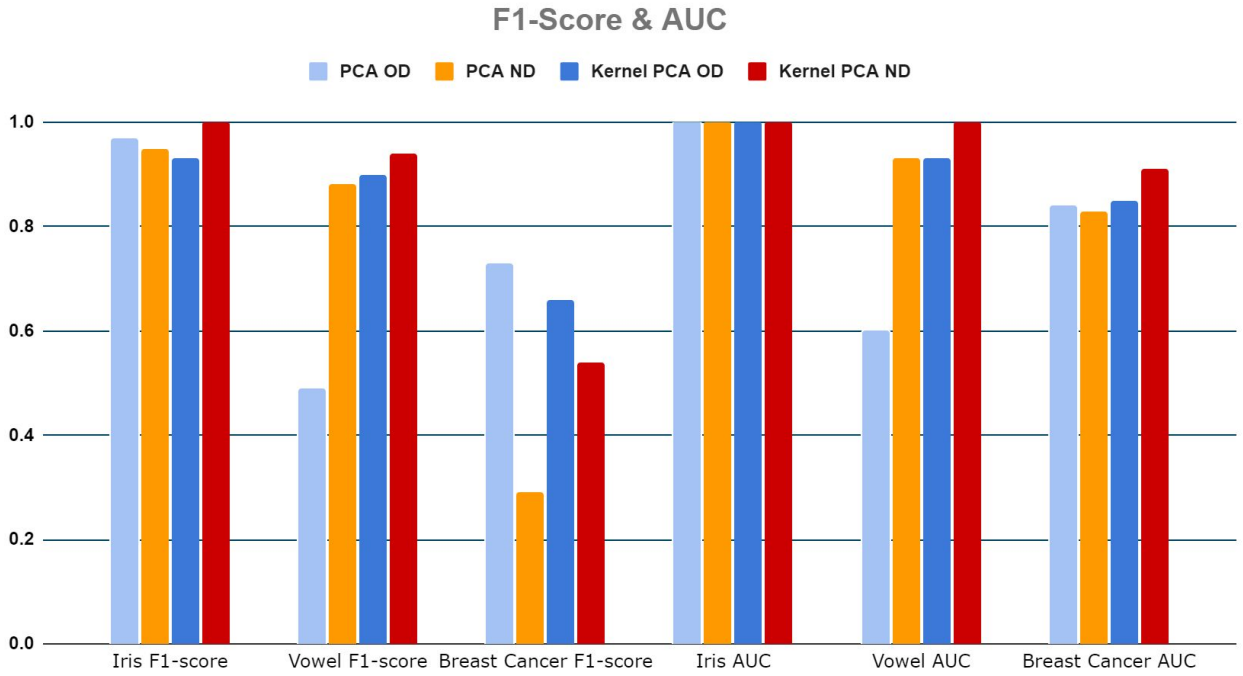
3. PCA: Number of components (I: 2, V: 6, BC: 7)
   Cumulative explained Variances with ~90%



4. KernelPCA: Fine-tuning Grid: {kernel,gamma}
   a. AUC
   b. F1-Score

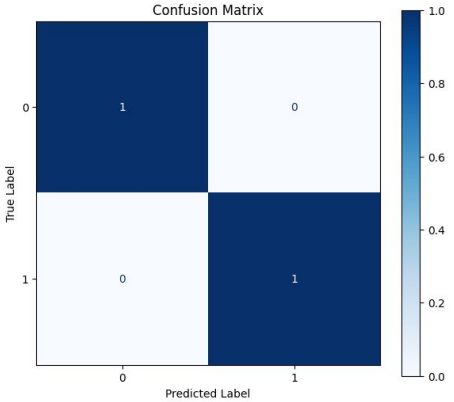
```
kernels = [ 'rbf',
            'poly'
            ]
gammas = np.logspace(-2,
                     2,
                     num=100,
                     base=10
                     )
```

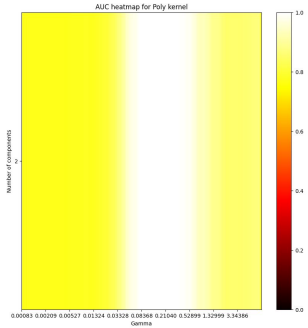
PCA and Kernel PCA

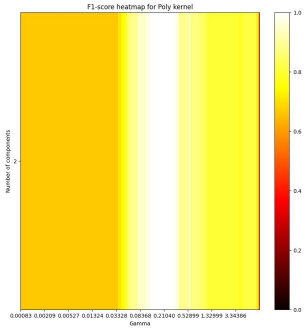


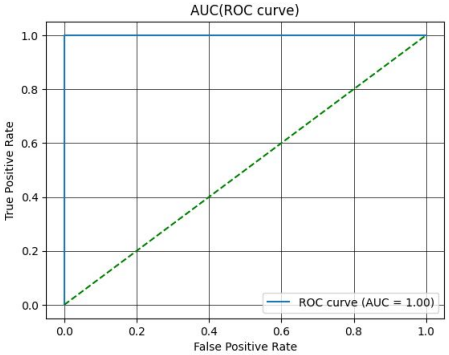
Kernel PCA ND on Iris Data Set

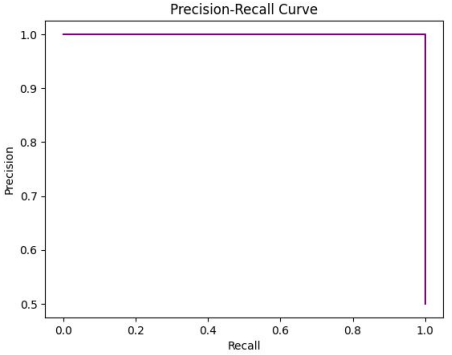
Best Performance		Optimum Parameters	
F1:	**1**	Kernel:	**poly**
AUC:	**1**	gamma:	**0.1256**
Accuracy	**1**		





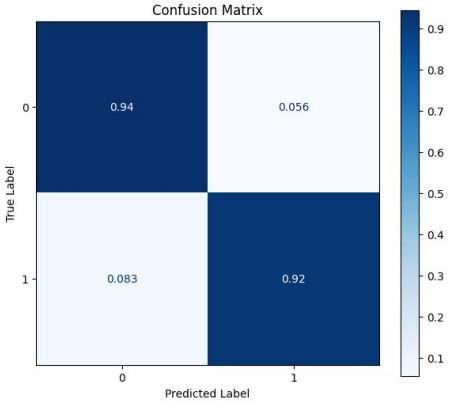


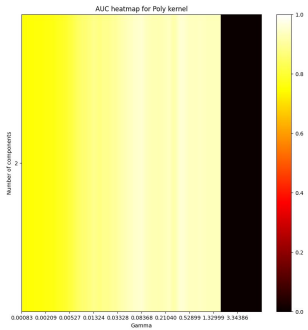


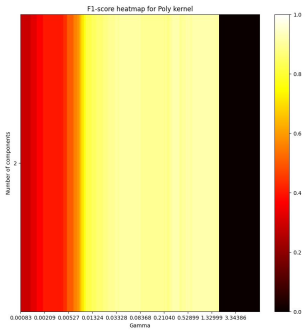


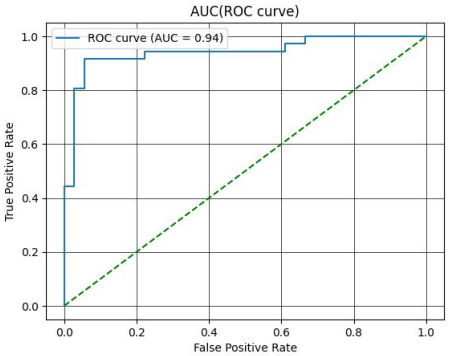
Kernel PCA ND on Vowels Data Set

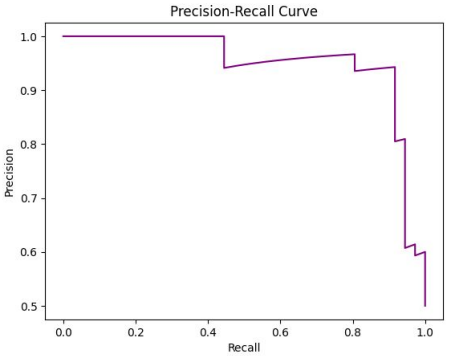
Best Performance		Optimum Parameters	
F1:	**0.9296**	Kernel:	**poly**
AUC:	**0.9421**	gamma:	**0.3367**
Accuracy	**0.9306**		





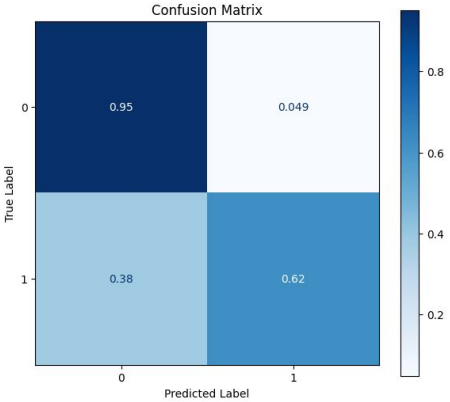


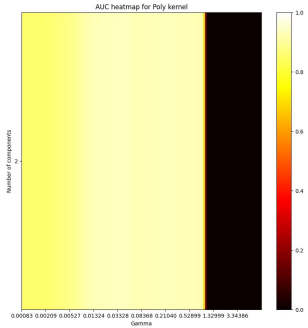


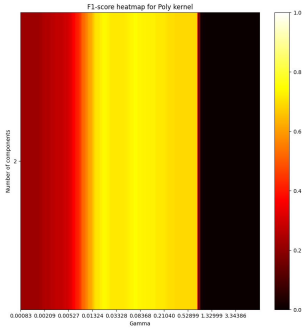


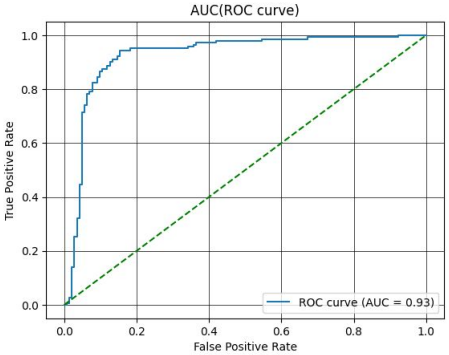
Kernel PCA ND on Breast Cancer Data Set

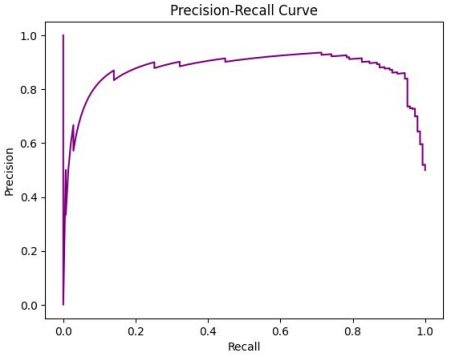
Best Performance		Optimum Parameters	
F1:	**0.7448**	Kernel:	**poly**
AUC:	**0.9251**	gamma:	**0.0671**
Accuracy	**0.7867**		





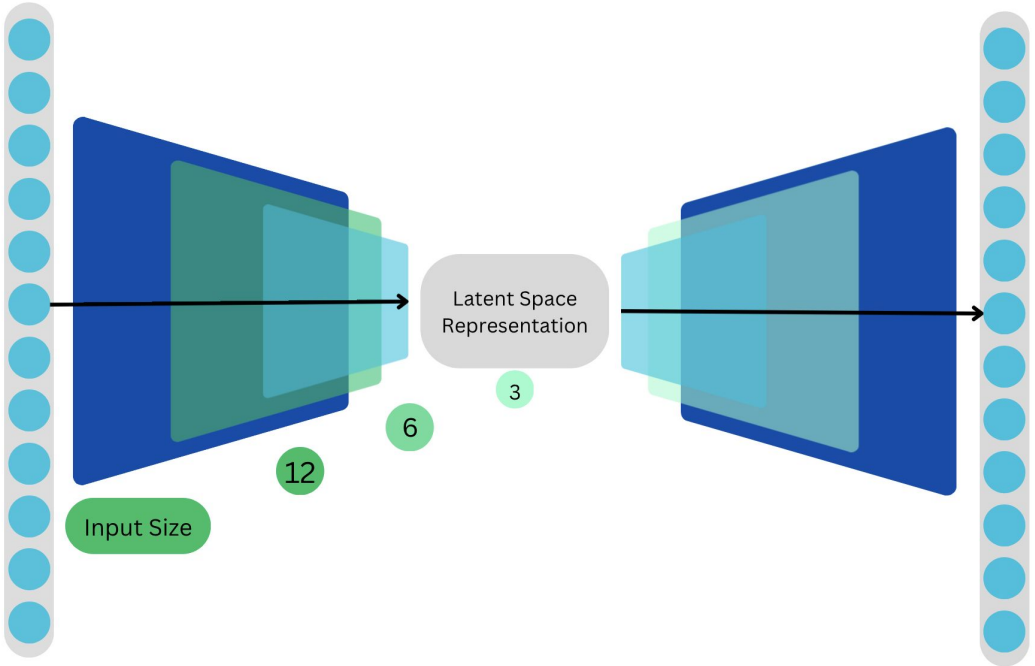




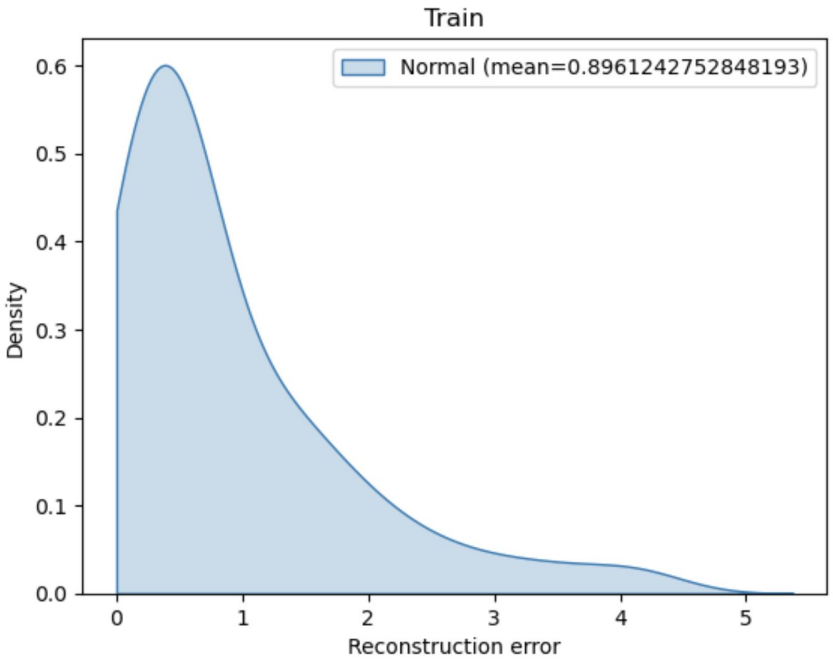


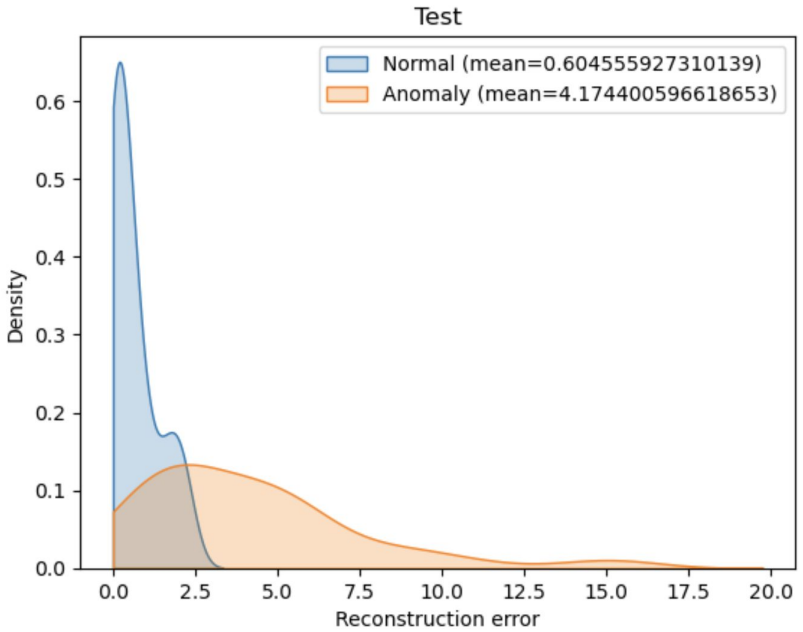
Fully Connected Auto-Encoder

- Six Linear Layers
- Twelve Hidden Nodes
- ReLu Activation Function
- Adam Optimizer



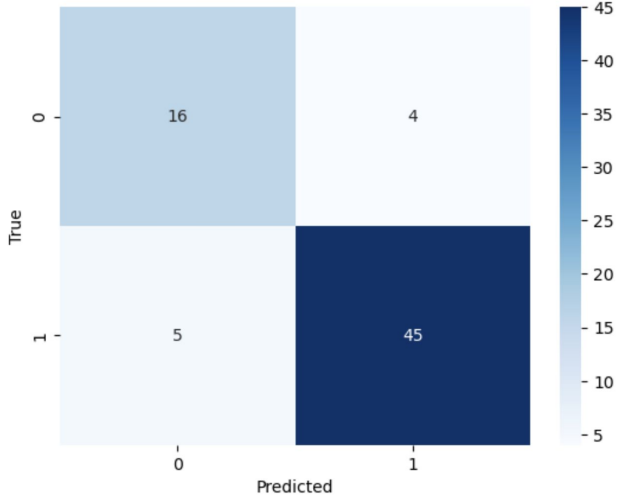
Results of Auto-Encoder on Iris Data Set

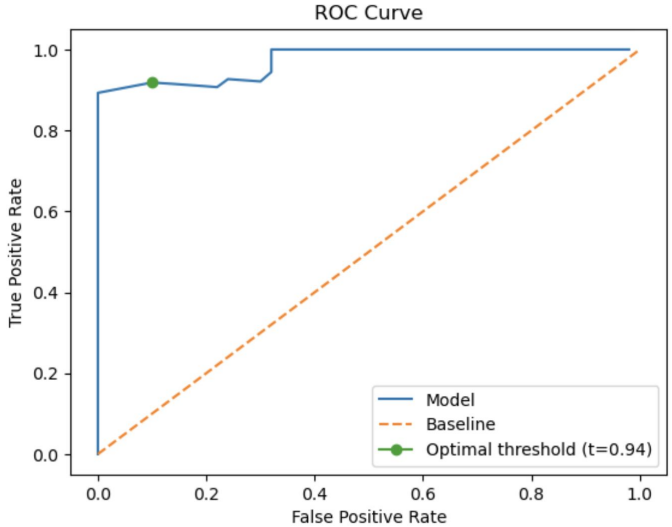


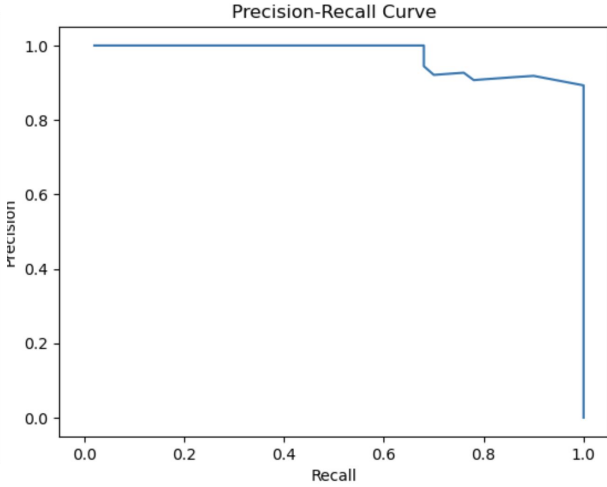


Results of Auto-Encoder on Iris Data Set

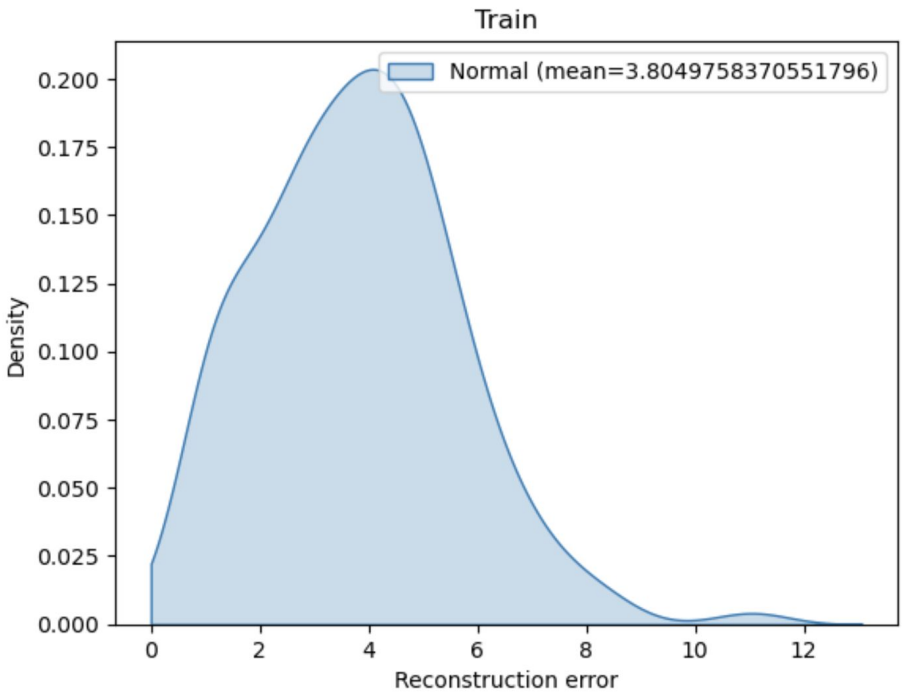
```
Accuracy: 0.871
Precision: 0.918
Recall: 0.9
F1: 0.909
J-Statistic: 0.936
```

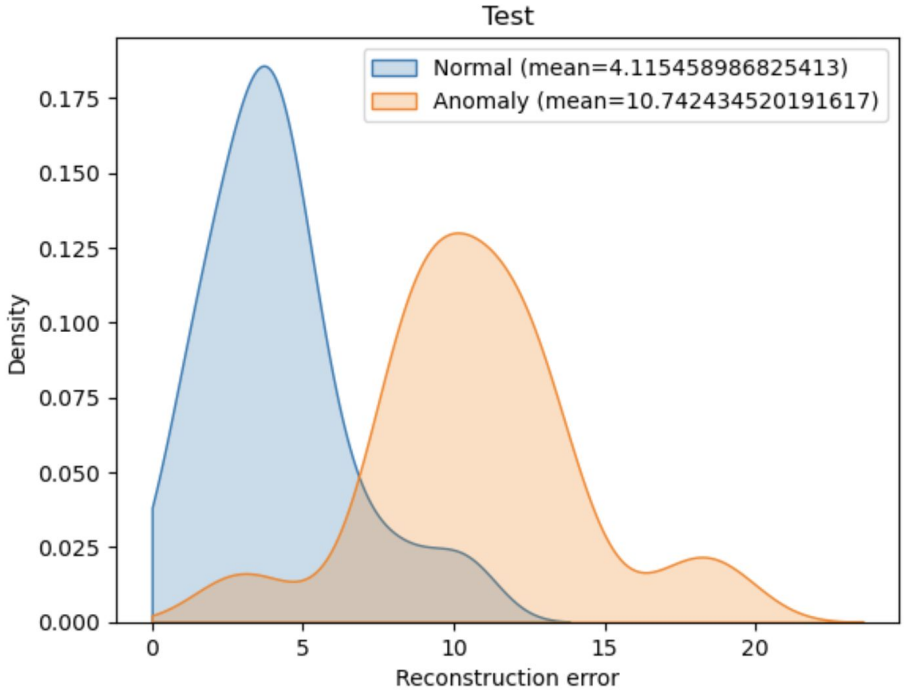






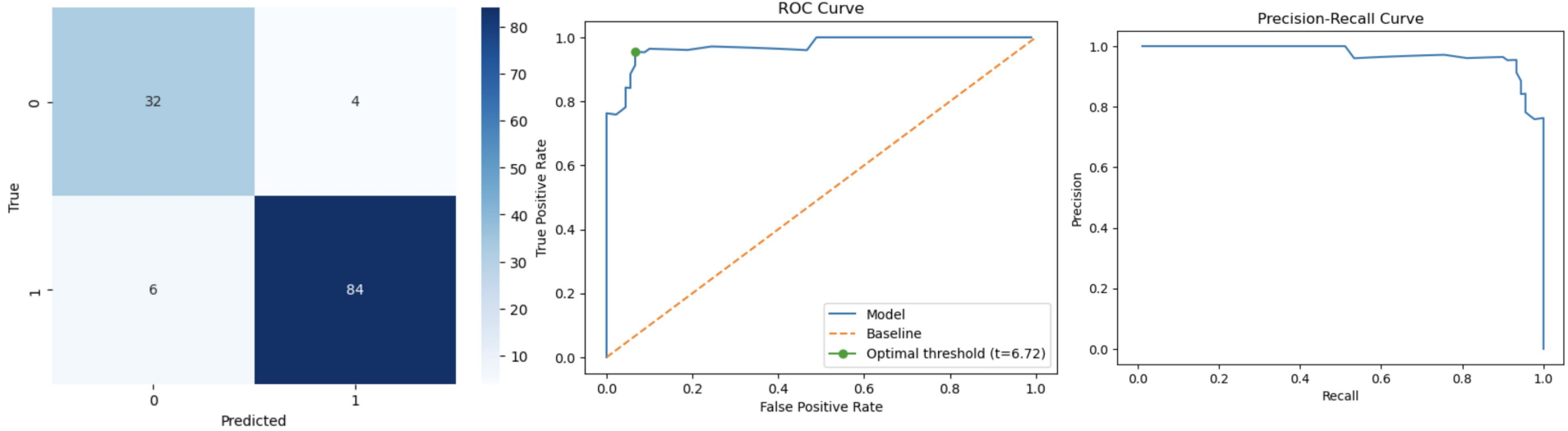
Results of Auto-Encoder on Vowel Data Set



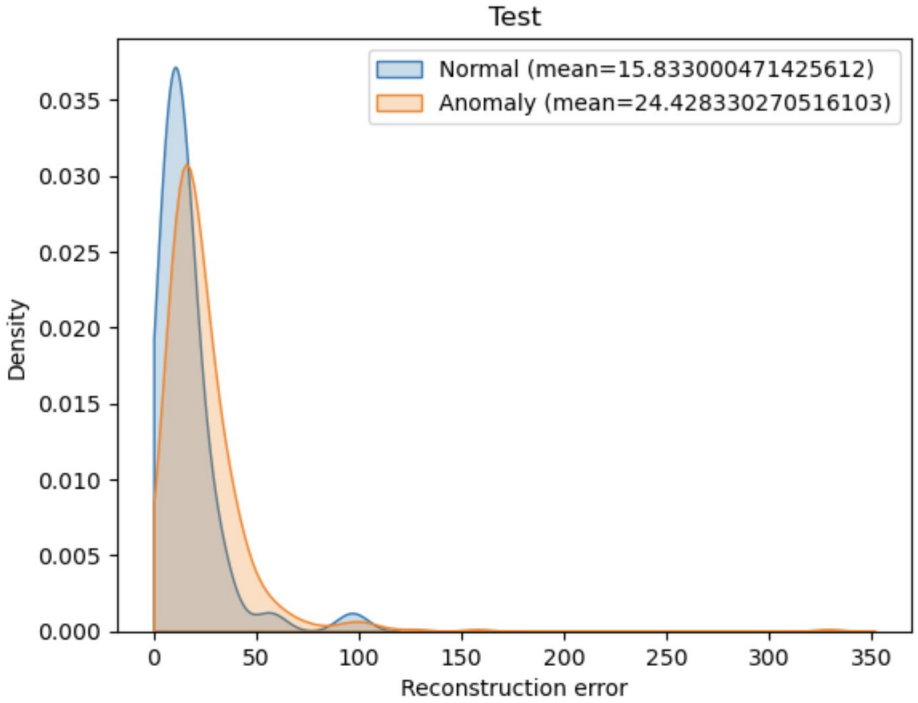


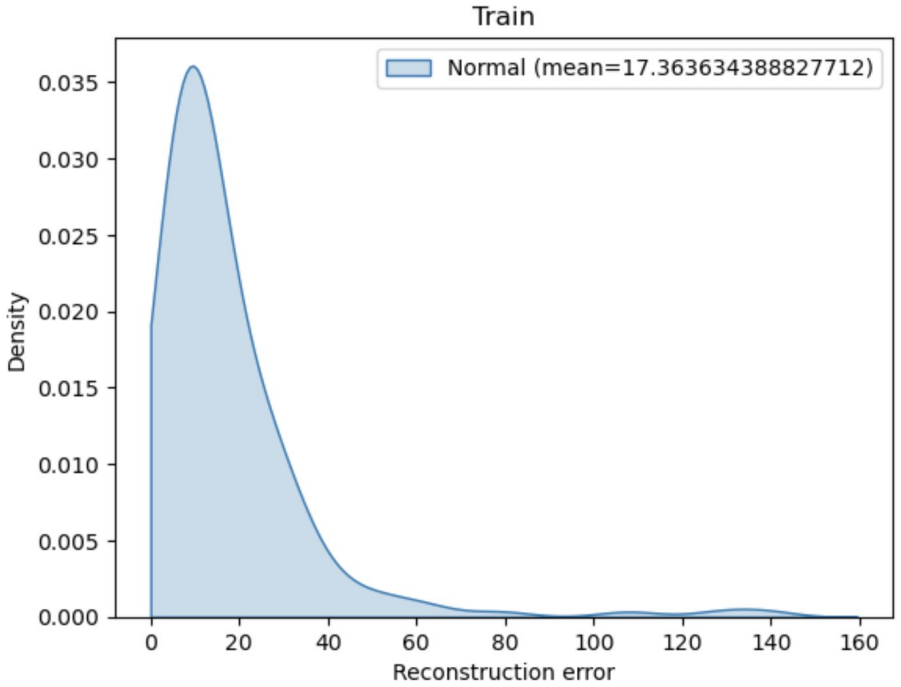
Results of Auto-Encoder on Vowel Data Set

```
Accuracy: 0.921
Precision: 0.955
Recall: 0.933
F1: 0.944
J-Statistic: 6.725
```



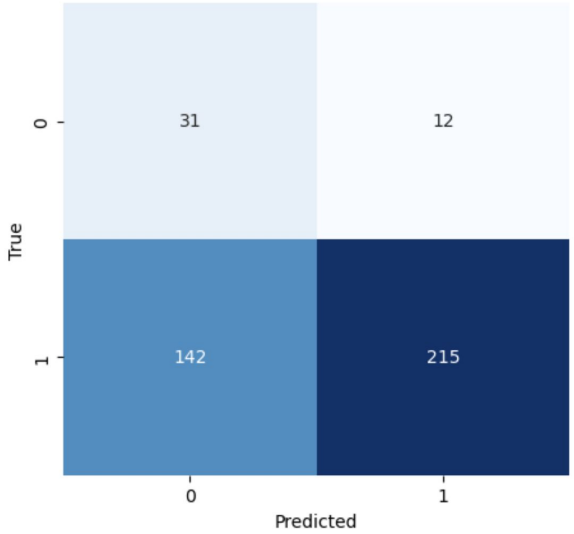
Results of Auto-Encoder on Breastcancer Data Set

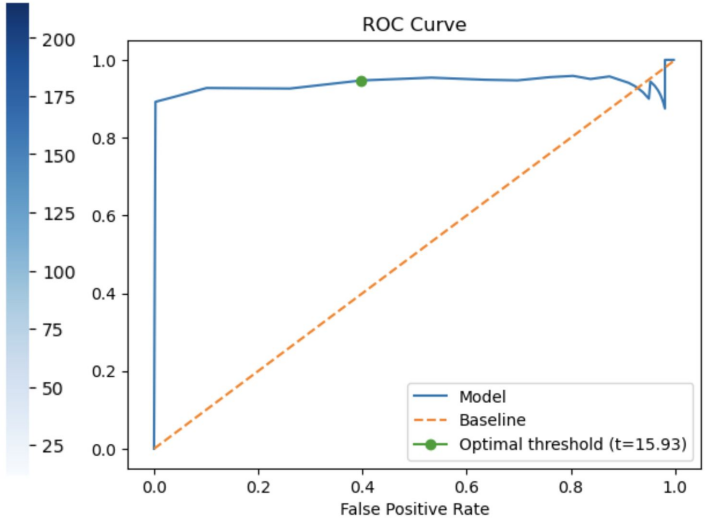


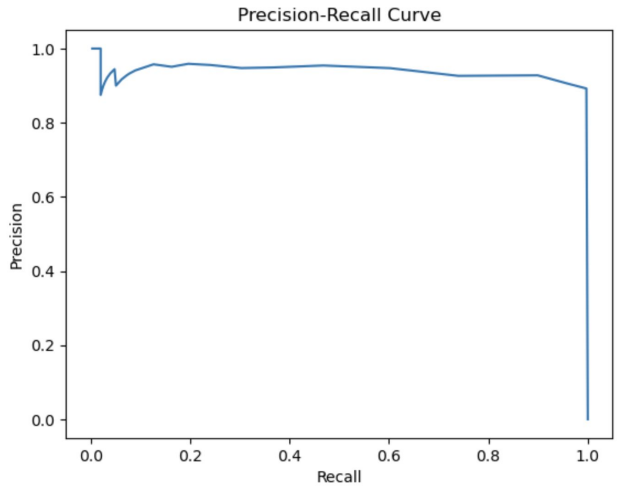


Results of Auto-Encoder on Breast cancer Data Set

```
Accuracy: 0.615
Precision: 0.947
Recall: 0.602
F1: 0.736
J-Statistic: 15.929
```



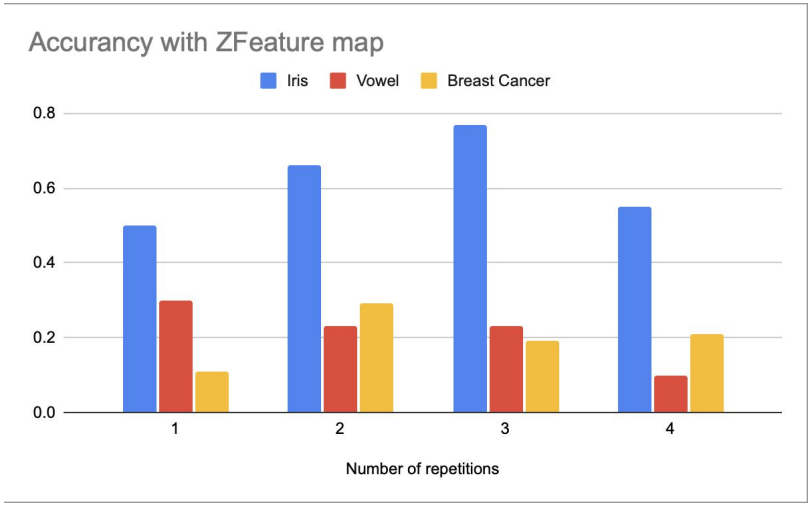


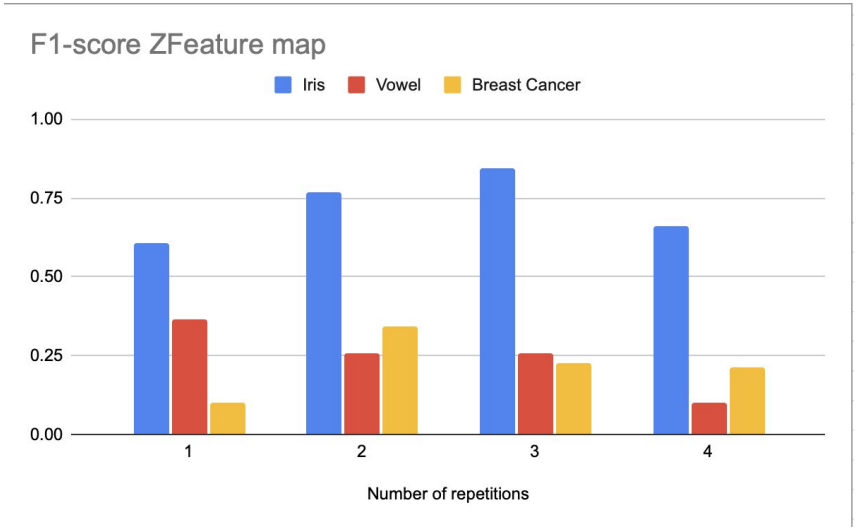


Quantum Implementation

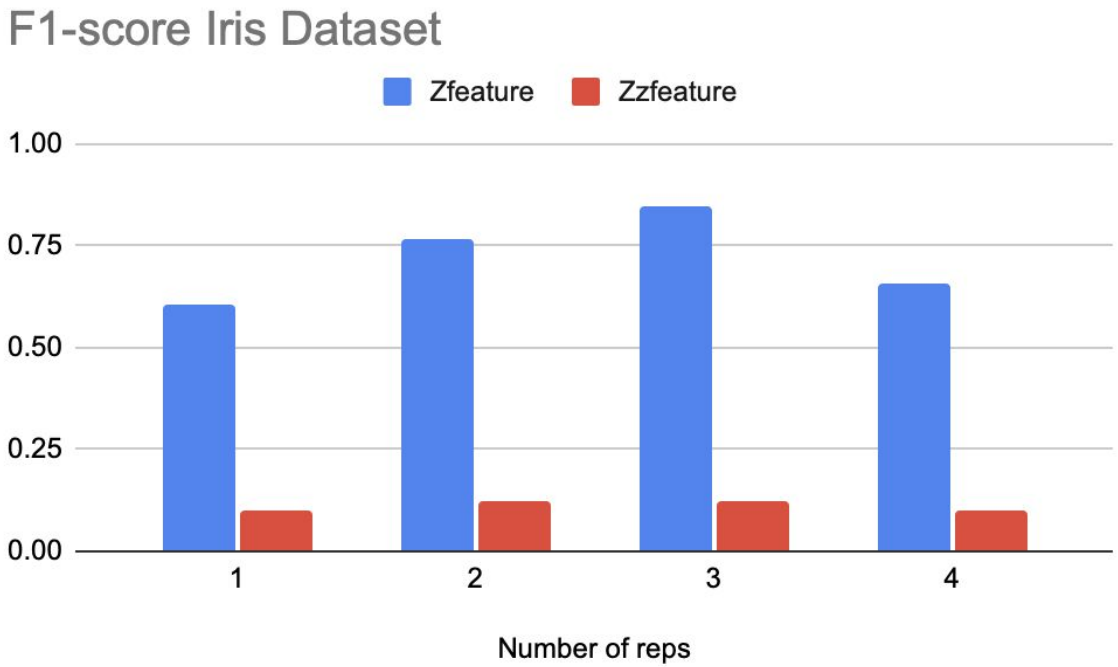
Quantum OCC- SVM

- Experiments were conducted on the QOCC-SVM by varying:
  1. Different feature maps for encoding : ZFeatureMap, ZZFeatureMap
  2. Number of reps



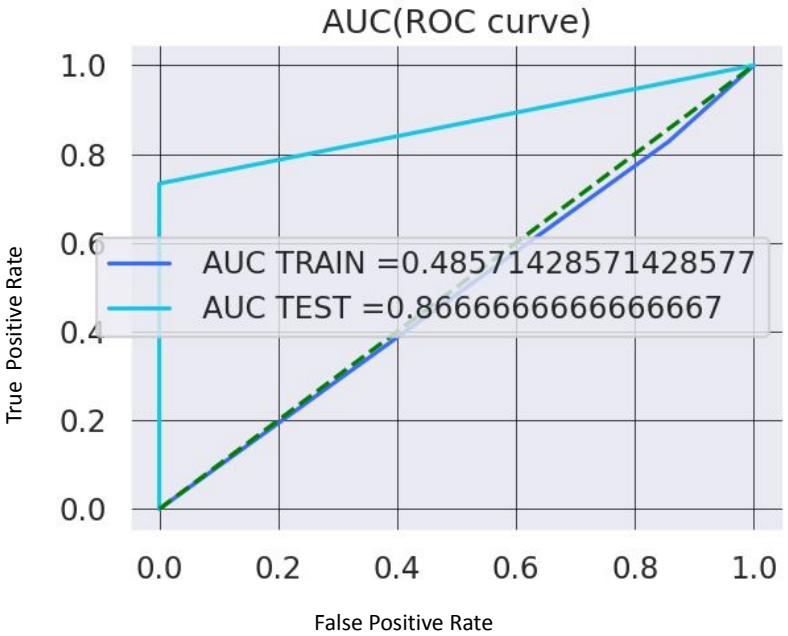


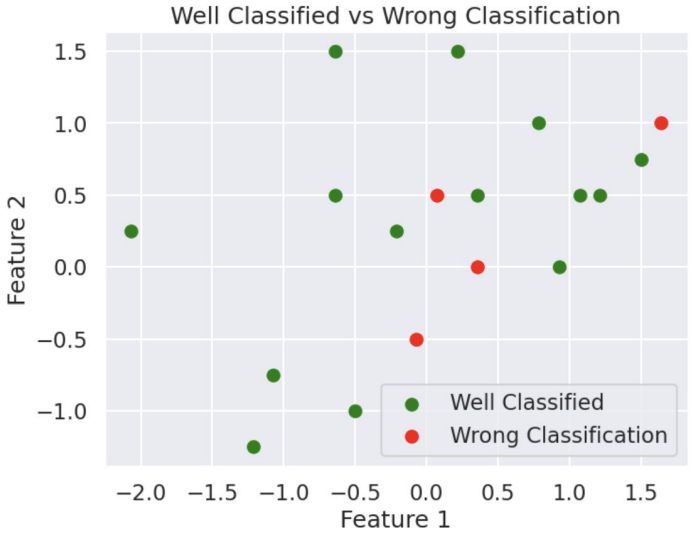
Zfeature map compared with ZZFeature map



Results of SVM-OCC on Iris Data Set

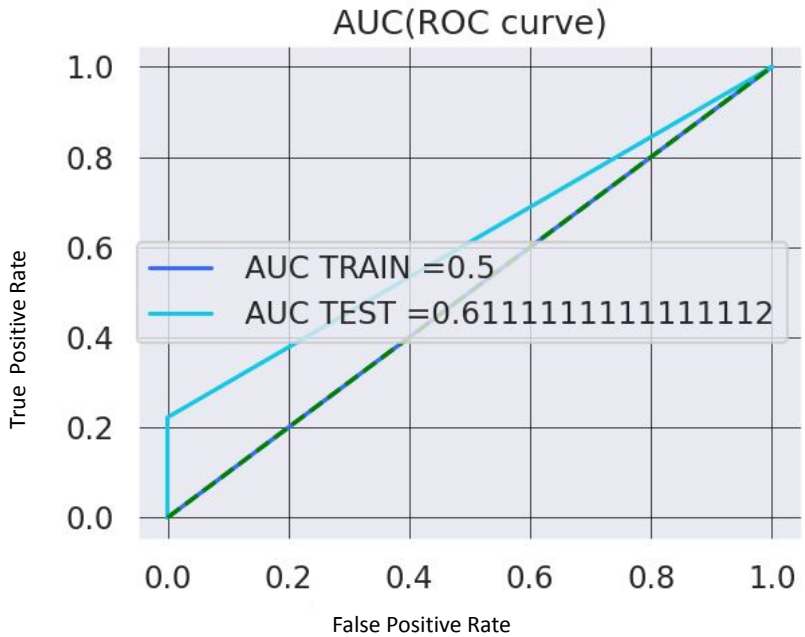
- Using ZFeatureMap with 3 reps

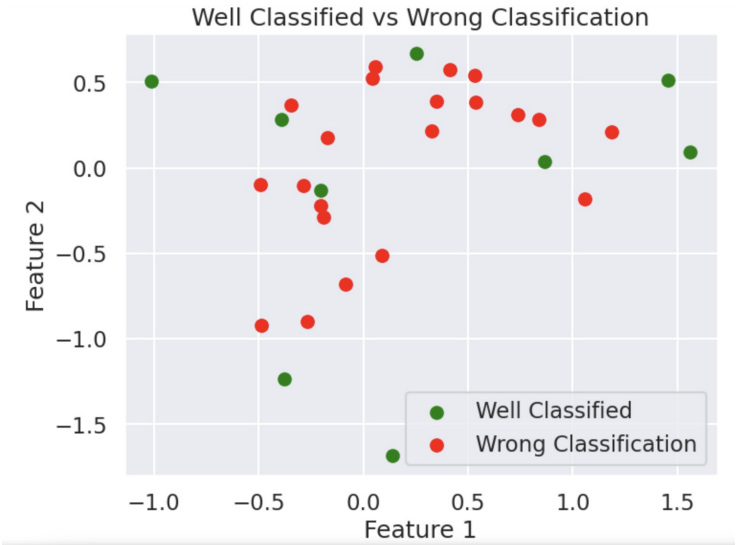




Results of SVM-OCC on Vowel Data Set

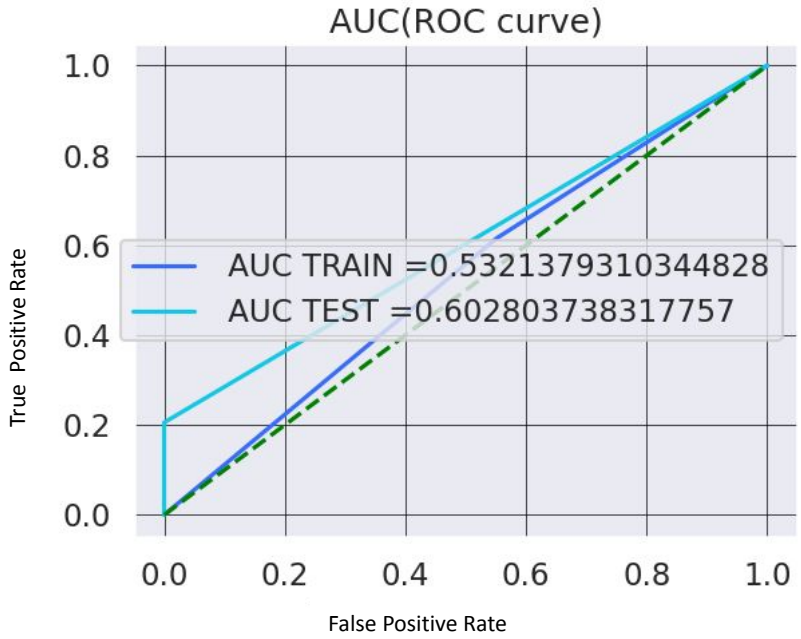
- Using ZFeatureMap with 1 reps





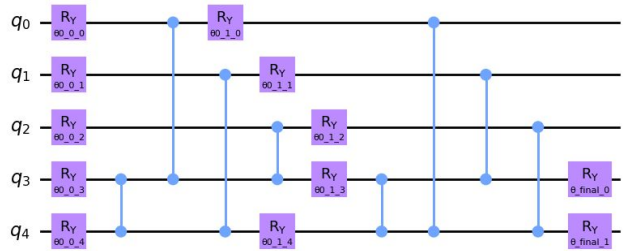
Results of SVM-OCC on Breast Cancer Data Set

- Using ZFeatureMap with 2 reps

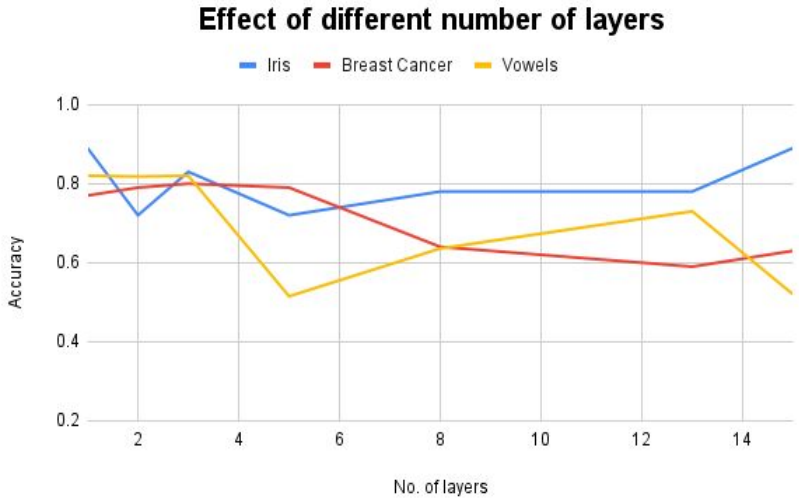


Variational Quantum Classifier as OCC

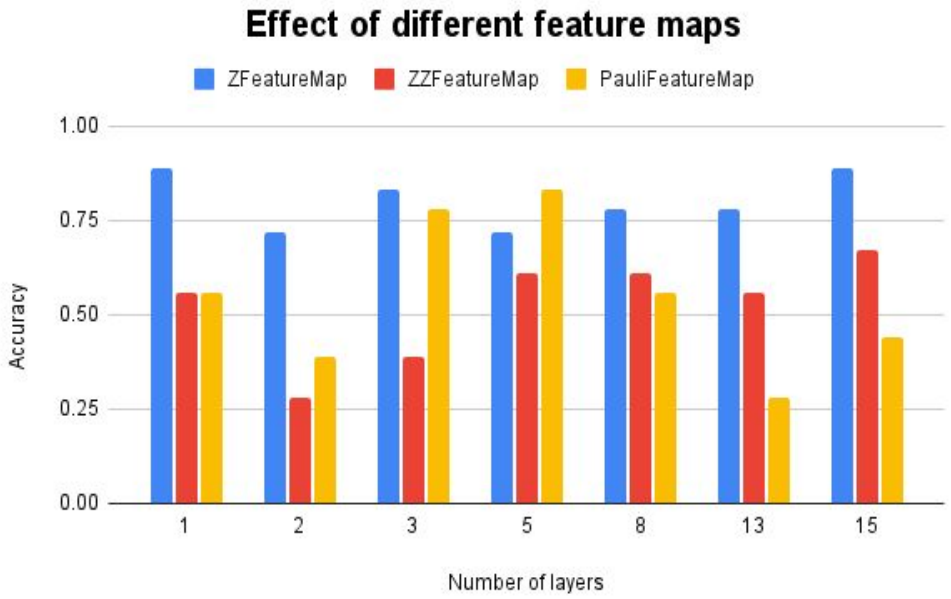
- Performed dimensionality reduction using PCA
- Different feature maps for encoding : ZFeatureMap, ZZFeatureMap and PauliFeatureMap
- Experiments were conducted on the VQC by varying:
  1. The number of layers in quantum circuit
  2. Using different optimizers

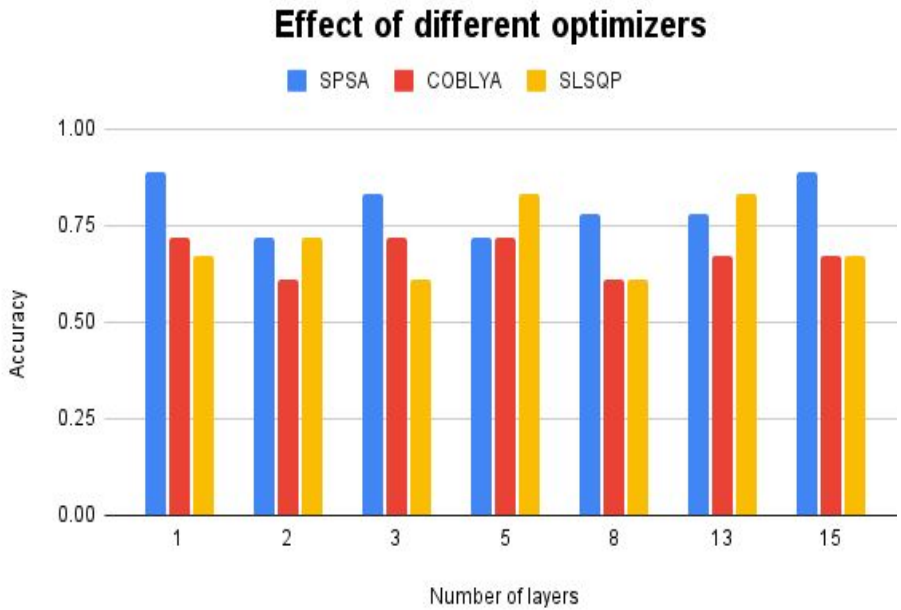


Implementation of QAE circuit: n_qubits: 5, t_qubits:2, layers:1



VQC - Effect of different feature maps and optimizers

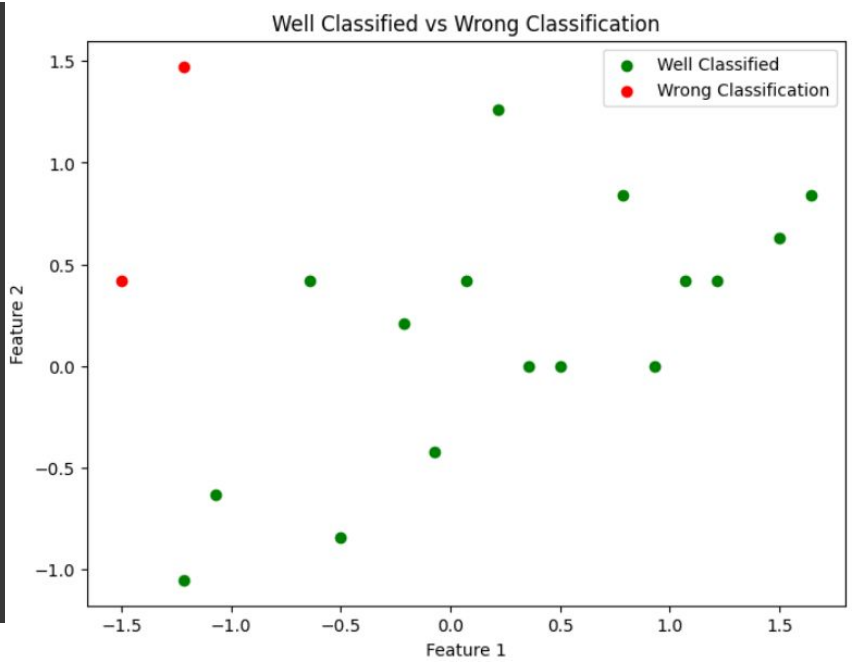




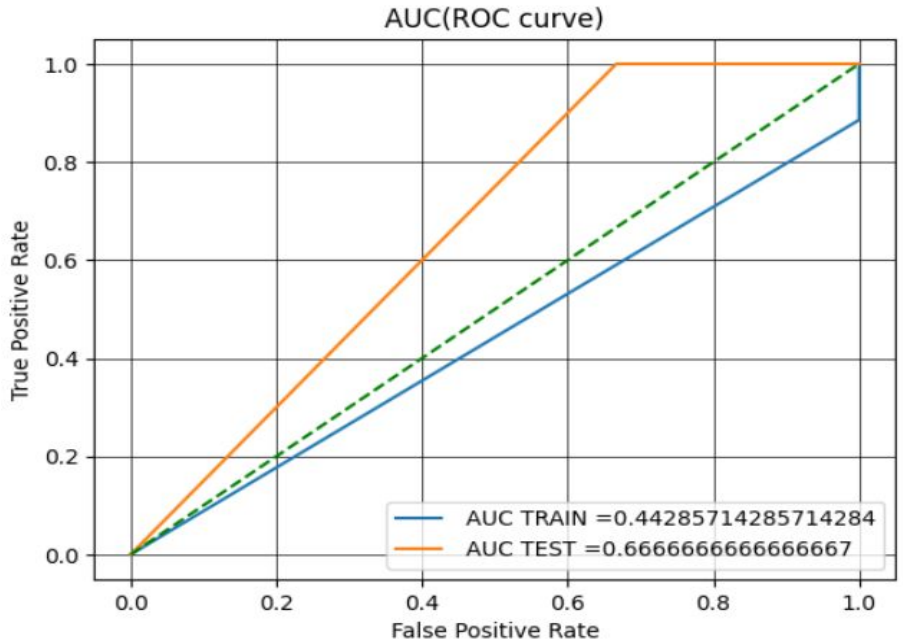
Results of VQCC on Iris Data Set

- VQC using ZFeatureMap having a single layer and using SPSA optimizer

```
Using 1 layers
Using feature map: ZFeatureMap
Using ansatz: RealAmplitudes
Using optimizer: SPSA
Time taken: 10.743 seconds
Testing accuracy: 0.889
Predicted:
 [1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1]
True:
 [1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1]
              precision    recall  f1-score   support

           0       1.00      0.33      0.50         3
           1       0.88      1.00      0.94        15

    accuracy                           0.89        18
   macro avg       0.94      0.67      0.72        18
weighted avg       0.90      0.89      0.86        18
```

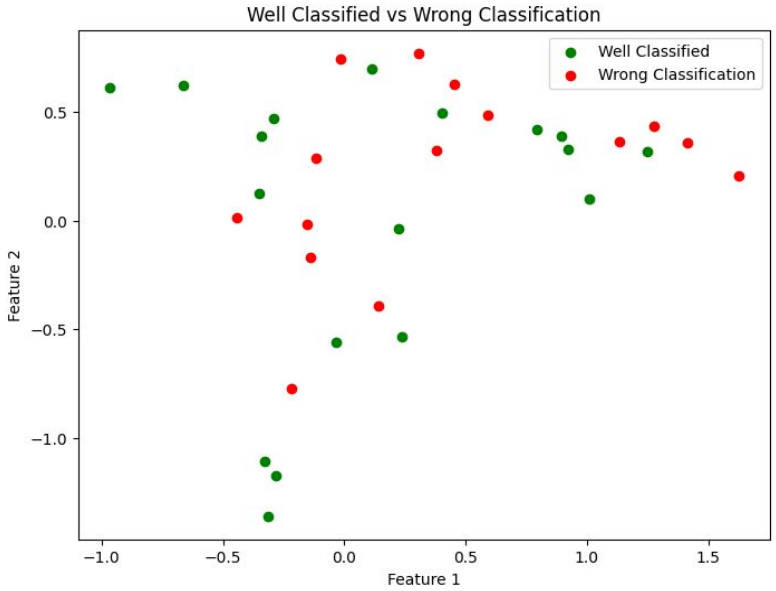


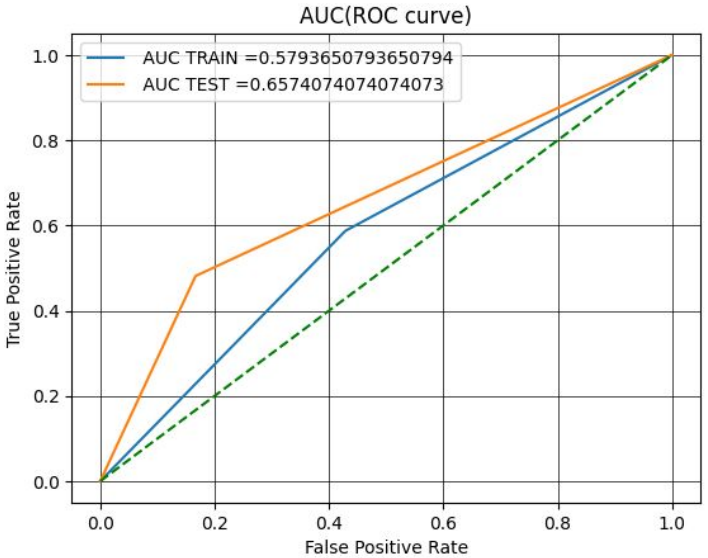
Results of VQCC on Iris Data Set



Results of VQC on Vowel Data Set

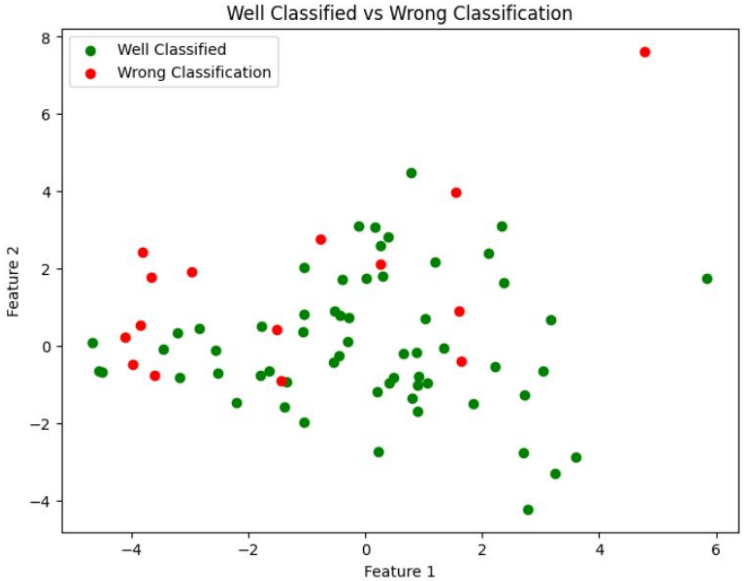
- VQC model with 3 layers using ZFeatureMap and SPSA optimizer predicted with an accuracy 0.82

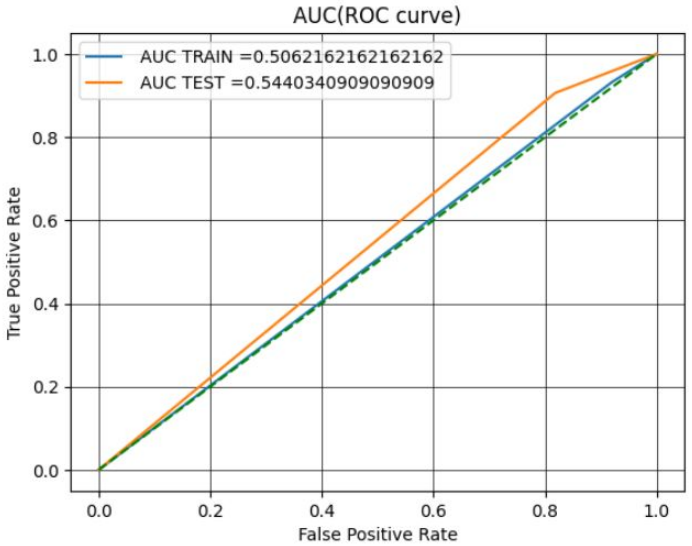




Results of VQC on Breast cancer Data Set

- VQC model with 3 layers using ZFeatureMap and SPSA optimizer predicted with an accuracy 0.8





Answer Research Questions

1. How do **different datasets affect** the performance of a **quantum one class classification method**?

2. How does the **quantum approach** perform in **comparison** to **classical methods** depending on the **number of parameters**?

3. How do **quantum one-class classification methods** compare to c**lassical one-class classification methods** in terms of **performance** and **computational complexity**?

4. What **benefits** are apparent by using **quantum computing with one classification**?

Conclusions and Future work

Conclusions

- **Classical methods** may struggle when confronted with **complex datasets** when **compared** with **quantum methods**.
- It’s unrealistic to expect quantum methods to surpass classical methods in terms of performance with the **current resources available**
- **Quantum method**s hold promise for providing innovative solutions and improved performance in handling specific data types, tackling the limitations faced by classical algorithms with advancements in quantum technology and implementation.

Future work

- Deeper analysis of parameters for OCCSVM
- Implementation of VQOCC
- Revise the approach for dataset selection

THANK YOU FOR YOUR ATTENTION

Maastricht University

References

[1] Tax, D.M., & Duin, R.P. (2006). Characterizing one-class datasets.
[2] Hoffmann, H. (2007). Kernel PCA for novelty detection. Pattern recognition, 40(3), 863-874.
[3] Perera, P., Oza, P., & Patel, V. M. (2021). One-class classification: A survey. arXiv preprint arXiv:2101.03064.
[4] Tax, D. M. J. (2002). One-class classification: Concept learning in the absence of counter-examples. 0584-0584.
[5] PyOD Documentation Release 1.0.9 by Yue Zhao
[6] Maheshwari, D., Sierra-Sosa, D., & Garcia-Zapirain, B. (2021). Variational quantum classifier for binary classification: Real vs synthetic dataset. IEEE Access, 10, 3705-3715.
[7] Park, G., Huh, J., & Park, D. K. (2022). Variational quantum one-class classifier. Machine Learning: Science and Technology.
[8] Liu (2018) Quantum machine learning for quantum anomaly detection
[9] Quantum feature maps and kernels https://learn.qiskit.org/course/machine-learning/quantum-feature-maps-kernels

Appendix

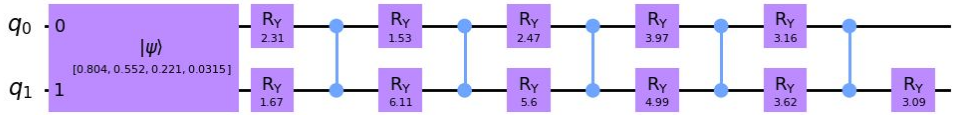
Implementation of VQOCC: (example on Iris Data Set)

```python
def QAE_circuit(params, nqubits, ntrash, layers, nparams):
    '''
    Create a Quantum Autoencoder Circuit with given parameters
    '''
    circuit = QuantumCircuit(nqubits)
    if (ntrash <= nqubits/2):
        for l in range(layers):
            for idx in range(ntrash):
                for q in range(nqubits):
                    #phase rotation
                    circuit.append(RYGate(params[q+idx*nqubits+l*ntrash*nqubits]), [q])
                # CZ between trash qubits
                for i,j in combinations(range(nqubits-ntrash,nqubits),2):
                    circuit.append(CZGate(), [i,j])
                # CZ between trash and non-trash qubits
                for i in range(ntrash):
                    for j in range(i,nqubits-ntrash,ntrash):
                        circuit.append(CZGate(), [nqubits-ntrash+((idx+i)%ntrash),j])
    else:
        for l in range(layers):
            for idx in range(nqubits-ntrash):
                for q in range(nqubits):
                    #phase rotation
                    circuit.append(RYGate(params[q+idx*nqubits+l*(nqubits-ntrash)*nqubits]), [q])
                # CZ between trash qubits
                for i,j in combinations(range(nqubits-ntrash,nqubits),2):
                    circuit.append(CZGate(), [i,j])
                for i in range(nqubits-ntrash):
                    for j in range(nqubits-ntrash+i,nqubits,nqubits-ntrash):
                        circuit.append(CZGate(), [(idx+i)%(nqubits-ntrash),j])
    for q in range(ntrash):
        circuit.append(RYGate(params[nparams-ntrash+q]), [nqubits-ntrash+q])

    return circuit
```

```python
# Amplitude encoding
def amplitude_encoding(data):
    # Ensure the data can be represented as a quantum state
    assert(np.abs(np.linalg.norm(data)-1) < 1e-3)
    n_qubits = int(np.log2(data.shape[0]))
    qc = QuantumCircuit(n_qubits)
    qc.initialize(data.tolist())
    return qc
```

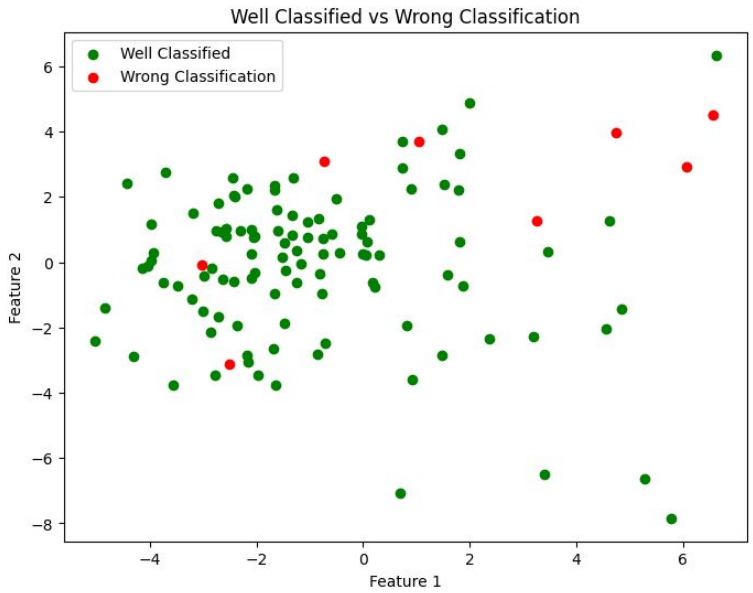
```
q_0: ─|0>─┐0
          │  State Preparation(0.80377,0.55161,0.22064,0.031521)
q_1: ─|0>─┘1
```

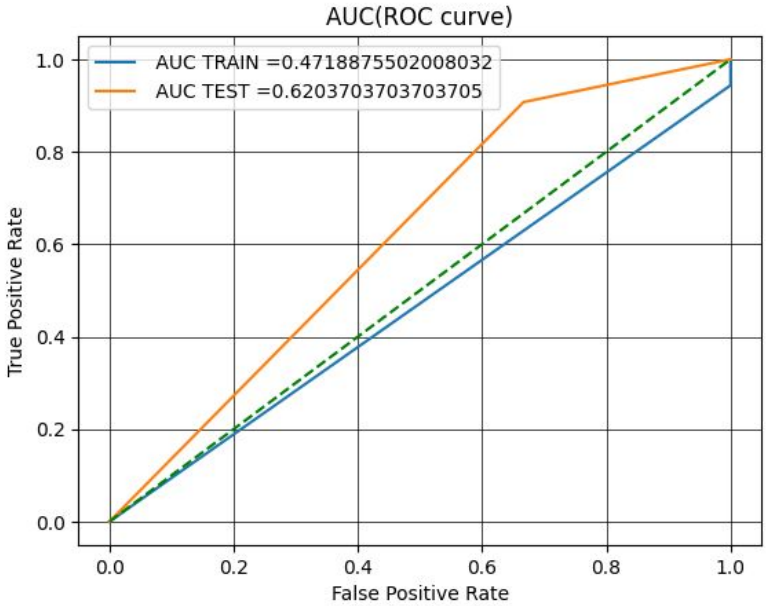
```python
nqubits, ntrash, layers = 2,1,5

if ntrash <= nqubits/2:
    nparams = ntrash * (nqubits * layers + 1)
else:
    nparams = (nqubits - ntrash) * nqubits * layers + ntrash
params = np.random.uniform(0,2*np.pi,nparams)
```



Appendix

VQC results with adhoc dataset:





Appendix

Kernel PCA on Breast Cancer Data Set for another attempt

Best Performance		Optimum Parameters	
F1:	**0.6154**	Kernel:	**poly**
AUC:	**0.9034**	gamma:	**0.2028**
Accuracy	**0.7222**