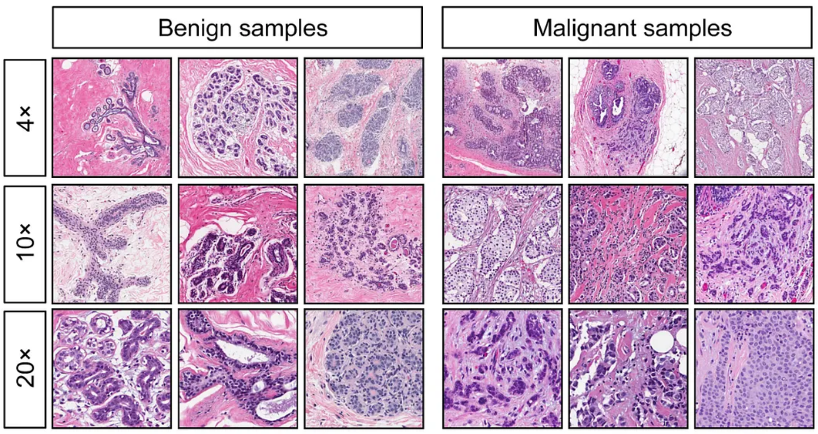
Quantum Computing and Machine Learning (614551008)

Practice 1.2 (2025-2026) Quantum Neural Networks

INSTRUCTIONS:

- **Deadline:** December 19th, 23:59.
- **Objectives**
  - In this practice we will develop a quantum neural network model for the Breast Cancer Wisconsin dataset.
- **Dataset**



- "**Breast Cancer Wisconsin**" is a dataset that contains measurements of cell nuclei extracted from digitized images of breast masses.
- It includes **569 samples**, each described by **30 numerical features** that capture properties such as radius, texture, perimeter, area, smoothness, compactness, and symmetry, computed both as mean values and as variations within each sample.
- Each instance is labeled as either **malignant or benign**, providing a clean and well-balanced target for supervised learning.

- Because the features are continuous and moderately correlated, the dataset is especially suitable for tasks involving dimensionality reduction (like PCA), visualization, and testing classical and quantum machine-learning algorithms.
- The dataset can be easily downloaded from *scikit-learn* using the following instructions:

```python
from sklearn.datasets import load_breast_cancer
data = load_breast_cancer()
X = data.data    # Features
y = data.target # Labels
```

- **Tasks to be carried out**

1. **Preprocessing**.
   - Preprocess the dataset to prepare it to feed the quantum neural network: Since all the data is numeric we have only to normalize then using, for example, the `MinMaxScaler` or the `StandardScaler` from `scikit-learn`.
   - Reduce the dimensionality using the PCA reduction from `scikit-learn` to reduce the number of qubits needed to represent the dataset in a quantum computer.
   - Divide the data into train and test, using, for example, the `train_test_split()` method from `scikit-learn`.
2. **Develop a quantum neural network to predict class of each case**.
   - Choose a quantum feature map to represent the wine data from the data encoding circuits available in Qiskit. (`https://quantum.cloud.ibm.com/docs/en/api/qiskit/circuit_library#data-encoding`)
   - Choose an ansatz (the classifier circuit) from the circuit library (`https://quantum.cloud.ibm.com/docs/en/api/qiskit/circuit_library`), e.g. `TwoLocal` or `RealAmplitudes`.
   - Choose an optimizer from the several available in `https://qiskit-community.github.io/qiskit-algorithms/apidocs/qiskit_algorithms.optimizers.html`
   - Build a Variational Quantum Classifier (VQC) using this feature map, ansatz and optimizer.
3. **Run the VQC with the train data**.
   - Optimizing the parameters to minimize the loss function.
4. **Results**.
   - Run the experiment several times with different combinations of feature map, ansatz and optimizer.
   - Compare the results obtained.
   - Use different metrics in addition to accuracy, keeping in mind that detecting a malignant case is more important than detecting a benign case.
5. **Conclusions**.
   - Reach some final conclusions for the experiment carried out.

- **Submission**
  - The exercises will be developed using Jupyter Notebooks.
  - **The notebook should include**:
    - The practice can be carried out alone or in pairs, so the first cell of the notebook must be the full names of the authors.
    - Include all the code developed.
    - The code shall be accompanied by cells with an explanatory report containing a description of the process followed, detailing the results obtained and justifying the decisions taken.
    - The notebook will be saved with the results of its execution included.
  - **Submission process**
    - The exercises will be submitted to professor (Eduardo Mosqueira Rey) by email (`mailto:eduardo.mosqueira@udc.es`) or by any other means agreed upon with him.
    - There is a strict deadline for each assignment. Past due submissions will be rejected.