## Variational Quantum Classifier

The variational quantum classifier (VQC) is a variational algorithm where the measured bitstrings are interpreted as the output of a classifier. Constructs a quantum circuit and corresponding neural network, then uses it to instantiate a neural network classifier.

## Exploratory Data Analysis

Exploratory Data Analysis (EDA) refers to the critical process of performing initial investigations on a dataset so as to discover patterns, spot anomalies, test hypotheses and check assumptions. It usually good practice in data science to explore the data first before getting your hands dirty and starting to build models.

## Steps in training a model using VQC

1. **Data Encoding :** This is also called data embedding. Data embedding is representation of classical data in quantum in Quantum systems. This step helps us translate our classic data to Hilbert space.
2. **State Preparation :** We will initialize the qubits and initialize them in initial state.
3. **Model the circuit :** We model circuit by constructing series of parameterized Rotation gates or Unitary Gates which evolves the input state. Parameters are external parameters which will be adjustable. The series of parameterized gates are called layers. Increasing layers introduce more trainable parameters into the model.
4. **Measurement :** The final step is the measurement step, which estimates the probability of belonging to a class by performing certain measurements. It's the equivalent of sampling multiple times from the distribution of possible computational basis states and obtaining an expectation value.

### Training

Training aim to find the values of parameters to optimize a given loss function. We can perform optimization on a quantum model similar to how it is done on a classical neural network. In both cases, we perform a forward pass of the model and calculate a loss function. We can then update our trainable parameters using gradient-based optimization methods since the gradient of a quantum circuit is possible to compute. During training we use the mean squared error (MSE) as loss function. This allows us to find a distance between our predictions and the truth, captured by the value of the loss function.

### Cost function

In supervised learning, the cost function is usually the sum of a loss function and a regularizer. We restrict ourselves to the standard square loss that measures the distance between target labels and model predictions.

## Implementing a VQC for fitting Parity Function

In this example I will explain all the steps for fitting Parity function referring to parityfunction.ipynb.

1. **Imports**
   Import pennylane, numpy and matplotlib for graphs.

2. **Quantum Node**

   a. Create a quantum device on which we can our quantum circuit.
   b. We define a layer which is an elementary block. We repeat these layers to fully build the variational circuit. Each layer consists of rotation gates and CNOT gates that entangle with neighboring qubits.
   c. We pass layer weights to each layer which will be adjustable.

3. **State Preparation**

   We encode our inputs into x. Our requirement is to prepare qubit state in 1 or 0 based on input binary bit. The BasisState function provided by PennyLane is made to do just this. It expects to be a list of zeros and ones.

4. **Define Cost function**

   Define a cost function that return the mean squared error between predictions and labels.

5. **Optimization**

   We initialize weights and biases with random values. Define `NesterovMomentumOptimizer for optimizing the circuit.`

6. **`Training`**

   `Initialize number of qubits to 4, Because we four inputs in X. Initialize number of layers to 2. We can define many number of layers and number of trainable parameters increase with layers. Initialize batch size to 5. Batch size defines no of inputs to be processed in one batch. Initialize iterations to run the entire batch for those many iterations and update weights.`

7. **`Predictions`**

   `Define a method accuracy that accepts adjusted weights, prediction sample and actual output to calculate the accuracy of the model.`