Data Fetching and Preprocessing

1. Fetching Data: The Statlog German Credit Data is fetched from the UCI Machine Learning Repository using the ucimlrepo library.
2. Preprocessing Data: Categorical variables in the dataset are encoded to numerical values using LabelEncoder and the target variable y is also encoded to numerical values if necessary.

Genetic Algorithm for Quantum Support Vector Machine (QSVM):

- Initialization: The function gsvm.gsvm is called to perform a genetic algorithm to optimize the QSVM parameters.
- Parameters: The parameters for the genetic algorithm include the number of qubits, depth of the quantum circuit, number of parameters, and the dataset (features X and targets y).
- Evolution Process: The genetic algorithm evolves a population of candidate solutions over several generations to find the best set of parameters that optimize the classification accuracy.

Double-click (or enter) to edit

```
!pip install qiskit
!pip install deap
!pip install -U scikit-learn
```

```
Collecting qiskit
  Downloading qiskit-1.3.2-cp39-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (12 kB)
Collecting rustworkx>=0.15.0 (from qiskit)
  Downloading rustworkx-0.15.1-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (9.9 kB)
Requirement already satisfied: numpy<3,>=1.17 in /usr/local/lib/python3.11/dist-packages (from qiskit) (1.26.4)
Requirement already satisfied: scipy>=1.5 in /usr/local/lib/python3.11/dist-packages (from qiskit) (1.13.1)
Requirement already satisfied: sympy>=1.3 in /usr/local/lib/python3.11/dist-packages (from qiskit) (1.13.1)
Collecting dill>=0.3 (from qiskit)
  Downloading dill-0.3.9-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: python-dateutil>=2.8.0 in /usr/local/lib/python3.11/dist-packages (from qiskit) (2.8.2)
Collecting stevedore>=3.0.0 (from qiskit)
  Downloading stevedore-5.4.0-py3-none-any.whl.metadata (2.3 kB)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.11/dist-packages (from qiskit) (4.12.2)
Collecting symengine<0.14,>=0.11 (from qiskit)
  Downloading symengine-0.13.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (1.2 kB)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.0->qiskit) (1.17.0)
Collecting pbr>=2.0.0 (from stevedore>=3.0.0->qiskit)
  Downloading pbr-6.1.0-py2.py3-none-any.whl.metadata (3.4 kB)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from sympy>=1.3->qiskit) (1.3.0)
Downloading qiskit-1.3.2-cp39-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (6.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.8/6.8 MB 41.9 MB/s eta 0:00:00
Downloading dill-0.3.9-py3-none-any.whl (119 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 119.4/119.4 kB 8.1 MB/s eta 0:00:00
Downloading rustworkx-0.15.1-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.0/2.0 MB 49.0 MB/s eta 0:00:00
Downloading stevedore-5.4.0-py3-none-any.whl (49 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.5/49.5 kB 3.3 MB/s eta 0:00:00
Downloading symengine-0.13.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (49.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.7/49.7 MB 8.1 MB/s eta 0:00:00
Downloading pbr-6.1.0-py2.py3-none-any.whl (108 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 108.5/108.5 kB 8.0 MB/s eta 0:00:00
Installing collected packages: symengine, rustworkx, pbr, dill, stevedore, qiskit
Successfully installed dill-0.3.9 pbr-6.1.0 qiskit-1.3.2 rustworkx-0.15.1 stevedore-5.4.0 symengine-0.13.0
Requirement already satisfied: deap in /usr/local/lib/python3.11/dist-packages (1.4.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (from deap) (1.26.4)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (1.6.0)
Collecting scikit-learn
  Downloading scikit_learn-1.6.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (18 kB)
Requirement already satisfied: numpy>=1.19.5 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.26.4)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.13.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (3.5.0)
Downloading scikit_learn-1.6.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.5/13.5 MB 81.5 MB/s eta 0:00:00
Installing collected packages: scikit-learn
  Attempting uninstall: scikit-learn
    Found existing installation: scikit-learn 1.6.0
    Uninstalling scikit-learn-1.6.0:
      Successfully uninstalled scikit-learn-1.6.0
Successfully installed scikit-learn-1.6.1
```

```
!git clone https://github.com/sergio94al/Automatic_design_of_quantum_feature_maps_Genetic_Auto-Generation.git
```

```
Cloning into 'Automatic_design_of_quantum_feature_maps_Genetic_Auto-Generation'...
remote: Enumerating objects: 1059, done.
remote: Counting objects: 100% (291/291), done.
remote: Compressing objects: 100% (134/134), done.
```

```
remote: Total 1059 (delta 180), reused 256 (delta 157), pack-reused 768 (from 1)
Receiving objects: 100% (1059/1059), 2.61 MiB | 13.79 MiB/s, done.
Resolving deltas: 100% (647/647), done.
```

```
import os
os.chdir('Automatic_design_of_quantum_feature_maps_Genetic_Auto-Generation')
```

```
pip install ucimlrepo
```

```
Requirement already satisfied: ucimlrepo in /usr/local/lib/python3.11/dist-packages (0.0.7)
Requirement already satisfied: pandas>=1.0.0 in /usr/local/lib/python3.11/dist-packages (from ucimlrepo) (2.2.2)
Requirement already satisfied: certifi>=2020.12.5 in /usr/local/lib/python3.11/dist-packages (from ucimlrepo) (2024.12.14)
Requirement already satisfied: numpy>=1.23.2 in /usr/local/lib/python3.11/dist-packages (from pandas>=1.0.0->ucimlrepo) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas>=1.0.0->ucimlrepo) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas>=1.0.0->ucimlrepo) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas>=1.0.0->ucimlrepo) (2024.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas>=1.0.0->ucimlrepo) (1.17.
```

```
import numpy as np
import pandas as pd
from ucimlrepo import fetch_ucirepo
from sklearn.preprocessing import LabelEncoder

def preprocess_statlog_german_credit_data():
    # Fetch dataset
    statlog_german_credit_data = fetch_ucirepo(id=144)

    # Data (as pandas dataframes)
    X = statlog_german_credit_data.data.features
    y = statlog_german_credit_data.data.targets.squeeze()  # Ensure y is a Series

    # Encode categorical variables
    categorical_columns = X.select_dtypes(include=['object']).columns
    label_encoders = {}
    for column in categorical_columns:
        label_encoders[column] = LabelEncoder()
        X[column] = label_encoders[column].fit_transform(X[column])

    # Convert target variable to numeric if necessary
    if y.dtype == 'object':
        y = LabelEncoder().fit_transform(y)

    return X.values, y

def evol(output="statlog_german_credit_result.csv"):
    X, y = preprocess_statlog_german_credit_data()  # Fetch the data using the defined function
    start = time.time()
    pop, pareto, logbook = gsvm.gsvm(nqubits=6, depth=6, nparameters=X.shape[1],
                                     X=X, y=y, weights=[-1.0,1.0],
                                     mu=50, lambda_=10, ngen=100)
    print(f'Simulation finished after {time.time()-start} seconds')
    print(f'\nGenetic-algorithm output ({output})')
    print('-------------------------------------------')
    with open(output, "w") as f:
        for ide, ind in enumerate(pareto):
            genes = ''.join(str(i) for i in list(ind))
            gates, acc = ind.fitness.values
            line = f'{ide},"{genes}",{gates},{acc}'
            f.write(line)
            f.write('\n')
            print(line)

# Run the evolution process
evol()
```

```
String: 10010100110001111000101101111001000101010111000100011010100001101101111000001100001011011100001010001001010111010000011101100100010
 -> accuracy = 0.7, gates = 21.0
String: 01110101101001111011100111110100100110111000011110010101101010001101010010110010101001001110110110010110010011101100100011101010011101010
 -> accuracy = 0.7, gates = 21.0
String: 11010101110101111011010011010101100110011101010110010101101011011011011111111010100110111011001011010001011110110010001011101001110001010
 -> accuracy = 0.705, gates = 6.0
100     10      0.7657  0.0270742       0.795   0.7
Simulation finished after 244.33904838562012 seconds

Genetic-algorithm output (statlog_german_credit_result.csv)
-----------------------------------------------
0,"11010101110101111011010011010101100110011101010110010101101011011011011111111010100110111011001011010001011110110010001011101001110001010010
1,"11010101110101111011010011010101100110011101010110010101101011011011011110001010100110111011001011010001011110110010001011101001110001010010
2,"11010101110101111011010011010101100110011101010110010101101011011011011111111010100110111011001011010001011110110010001011101001110001010010
3,"11010101100000011011110110111011010101110101010010110110101011110101000111101010100110111011001011010001011110110010001011101001110001010010
4,"11010101110101111011110110111011110111001110101011000101011010110111011011110001010100110111011001011010001011110110010001011101001110001010010
5,"11010101110101111011110110111011110111001110101011000101011010110111011011110001010100110111011001011010001011110111010001011101001110001010010
6,"11010101110101111011110110111011110111001110101011001010110101101110110111100010101001101110110010110100010111101110100010111010011100010100
7,"11010101110101111011110110111011110111001110101011001010110101101110110111100010101001101110110010110100010111101100100010111010011100010100
8,"11010101110101111011110110111011110111001110101011000101011010110111011011110001010100110111011001011010001011110111010001011101001110001010010
9,"11010101110101111011110110111011110111001110101011000101011010110111011011110001010100110111011001011010001011110110010001011101001110001010010
10,"11010101110101111011101100111011110111001110101011001010110101101110110111100010101001101110110010110100010111101110100010111010011100010100
11,"11010101110101111011101100111011110111001110101011001010110101101110110111100010101001101110110010010100010111101100100010111010011100010100
12,"11010101110101111011101100111011110111001110101011001010110101101110110111100010101001101110110010110100010111101100100010111010011100010100
13,"11010101110101111011101100111011110111001110101011001010110101101110110111100010101001101110110010010100010111101110100010111010011100010100
14,"11010101110101111011101100111011110111001110101011001010110101101110110111100010101001101110110010110100010111101100100010111010011100010100
15,"11010101100000011011010011010111010101011101010110010101101010111101010001111110011001101110110010110100010111101100100010010101011011010100
16,"11010101100000011011101100111010110011001100101011001010110101011110101000111101010100110111011001001010001011110111010011011101001110001010
17,"11010101100000011011101100111010110011001100101011001010110101011110101000111101010100110111011001011010001011110111010011011101001110001010
18,"11010110110101111011101100111010110011001100101011001010110101011110101000111111001110110111011001011010001011110110010001011101001110001010
19,"10100110110101111011101100111010110011001100101011001010110101011110101000111111001110110111011001011010001011110110010001011101001110001010
20,"11010101100000011011101100111010110011001100101011001010110101011110101000111101010100110111011001001010001011110110010001011101001110001010
21,"11010101100000011011101100111010110011001100101011001010110101011110101000111101010100110111011001011010001011110110010011011101001110001010
22,"11010101100000011011101100111010110011001100101011001010110101011110101000111101010100110111011001011010001011110111010001011101001110001010
23,"11010101100000011011101100111010110011001100101011001010110101011110101000111101010100110111011001011010001011110110010001011101001110001010
24,"11010101100000011011101100111010110011001100101011001010110101011110101001111000101010011011101100100101000101111011010011011101001110001010
25,"11010101100000111011101100111010110011001100101011001010110101011110101001111000101010011011101100100101000101111011010001011101001110001010
26,"11010101100000011011101100111010110011001100101011001010110101011110101001111000101010011011101100100101000101111011010011011101001110001010
27,"11010101100000011011101100111010110011001100101011001010110101011110101001111000101010011011101100100101000101111011010011011101001110001010
```

Logging and Output:

- Recording Results: The results of the genetic algorithm, including the genes, number of gates, and accuracy for each individual in the Pareto front, are logged and written to a CSV file.
- Output: The results are printed to the console and saved in the specified output file.