ï»¿Part 1: Neural Networks for Classification

1. Determine and report the network architecture, including the number of input nodes, the number of output nodes, the number of hidden nodes (assume only one hidden layer is used here). Describe the rationale of your choice.

The number of input nodes is 4, because there are four numeric attributes.
The number of output nodes is 3, because there are three classes.
The number of hidden nodes is 2, when hidden nodes is 1, training rate is really low(36%) which means train is unsuccessful. Then the 2 is smallest number of hidden nodes and train is successful.

2. Determine the learning parameters, including the learning rate, momentum, initial weight ranges, and any other parameters you used. Describe the rationale of your choice.

The learning rate is 0.05. the range of learning rate from 0 to 1. because I want to get higher accuracy. But if the learning rate is really small
The momentum is 0.0 which is default value.
The initial weight ranges is 0.001. I set this value really small because I do not want randomness to effect the result.
The classification accuracy is 101.0. when classification accuracy more than 100, then the parameter is useless.
The number of layers is 3. there is 1 hidden layer here.

3. Determine your network training termination criteria. Describe the rationale of your decision.

My network training termination criteria is mean squared error. I set them as 0.001 and 101% respectively, which means even though the percent of classification accuracy has reached to 100%, the critical error must be less than 0.001. The practical result illustrates the termination criteria will improve the accuracy. I do not set mean squared error to 0, because I want to avoid the over training problem.

4. Report your results (average results of 10 independent experiment runs with different random seeds) on both the training set and the test set. Analyse your results and make your conclusions.

run 1
Mean squared error for training data: 0.001
Number of incorrect classifications: 0/75
Mean squared error for test data: 0.018
Number of incorrect classifications: 2/75
run 2
Mean squared error for training data: 0.001
Number of incorrect classifications: 0/75
Mean squared error for test data: 0.018
Number of incorrect classifications: 2/75
run 3
Mean squared error for training data: 0.001
Number of incorrect classifications: 0/75
Mean squared error for test data: 0.020
Number of incorrect classifications: 2/75
run 4
Mean squared error for training data: 0.001
Number of incorrect classifications: 0/75
Mean squared error for test data: 0.023
Number of incorrect classifications: 3/75

run 5
Mean squared error for training data: 0.001
Number of incorrect classifications: 0/75
Mean squared error for test data: 0.018
Number of incorrect classifications: 2/75
run 6
Mean squared error for training data: 0.001
Number of incorrect classifications: 0/75
Mean squared error for test data: 0.019
Number of incorrect classifications: 3/75
run 7
Mean squared error for training data: 0.001
Number of incorrect classifications: 0/75
Mean squared error for test data: 0.018
Number of incorrect classifications: 2/75
run 8
Mean squared error for training data: 0.001
Number of incorrect classifications: 0/75
Mean squared error for test data: 0.018
Number of incorrect classifications: 2/75
run 9
Mean squared error for training data: 0.001
Number of incorrect classifications: 0/75
Mean squared error for test data: 0.019
Number of incorrect classifications: 3/75
run 10
Mean squared error for training data: 0.001
Number of incorrect classifications: 0/75
Mean squared error for test data: 0.018
Number of incorrect classifications: 2/75
conclusion:
This method is stable and accurate. It is good classifier for appreciate parameters. Average of correct classifications approximately 97.33%.

5. (optional/bonus) Compare the performance of this method (neural networks) and the nearest neighbour methods.

k nearest neighbour algorithm in assignment 1. accuracy is 96% when k = 3. accuracy is 94.66667% when k = 5. Thus, according to the result of above. So the performance of neural networks(97.333%) is higher than nearest neighbour methods(96%).

Part 2: Genetic Programming for Symbolic Regression

1. Determine a good terminal set for this task.

terminal set for this task just variable of x input and constant from -4.0d to 4.0d.

2. Determine a good function set for this task.

My function set has Multiply, Divide, Subtract, Add, Pow which is only basically Math function. I did not add Sine and Cosine. Because it produce 0 some times. if number divide by zero will lead to error.

3. Construct a good fitness function and describe it using plain language (and mathematical formula, or other formats you think appropriate).

The sum of the differences between expected Y and actual Y is the fitness. Which is

abs value of the difference between the output calculated by the genotype and the true output every generation. If the result closer to zero, then it is better fitness set.

4. Describe the relevant parameter values and the stopping criteria you used.

Max initial depth: 4
Population size: 1000
Max Tree Depth: 8
Crossover Probability: 80%
Mutation Probability: 20%
Reproduction Probability: 5%
Stopping criteria: 2000 generations or overall fitness function equal 0.

which means that goes over the 1000 population and select the best result from that generation. Then it insert the result of the genetic operations into new generation until reach stopping criteria.

5. List three different best programs evolved by GP and the fitness value of them (you need to run your GP system several times with different random seeds and report the best programs of the runs).

Genotype 1:
Best solution fitness: 0.0
Best solution: ((1.0 - (0.0 - X)) + (((X * X) - X) * ((X * X) - X))) - X

Genotype 2:
Best solution fitness: 0.0
Best solution: ((X * X) + (X ^ 0.0)) + ((((X * X) - X) - X) * (X * X))

Genotype 3:
Best solution fitness: 0.0
Best solution: (((X * X) - X) ^ 2.0) + (4.0 / 4.0)

6.(optional, bonus) Analyse one of the best programs to reveal why it can solve the problem in the task.

Genotype2 is best: random select variable of x as 1.00 and -2.00, then put X variable to function 2. Then only genotpye2 get suitable result of Y which is 1.00 and 37.00.

Part 3: Genetic Programming for Classification

1. Determine a good terminal set for this task.

terminal set for this task just variable of 9 variable input and constant from -4.0d to 4.0d.

2. Determine a good function set for this task.

My function set has Multiply, Divide, Subtract, Add, Pow which is only basically Math function. I did not add Sine and Cosine. Because it produce 0 some times. if number divide by zero will lead to error.

3. Construct a good fitness function and describe it using plain language (and mathematical formula, or other formats you think appropriate).

A good fitness function is a low error rate or high accuracy. I chose high

accuracy. prediction works from tutorial is that if ProgOut < 0 then class1 else class2 below is my code:

```
public PatientFitnessFunction(List<Patients> cancers){
    this.cancers = cancers;
}
protected double evaluate(IGPProgram igpProgram) {
    double result = 0.0f;
    for (Patients patient : cancers) {
        problem.setPatientsVariable(patient);
        double value = igpProgram.execute_float(0, NO_ARGUMENTS);
        if (value >= 0) {
            if (patient.getCondition()==2) {
                result++;
            }
        } else {
            if (patient.getCondition()==4) {
                result++;
            }
        }
    }
    return result / cancers.size();
}
```

4. Describe the relevant parameter values and the stopping criteria you used.

Max initial depth: 4
Population size: 1000
Max Tree Depth: 8
Crossover Probability: 80%
Mutation Probability: 20%
Reproduction Probability: 5%
Stopping criteria: 2000 generations or overall fitness function less then 0.04.

which means that goes over the 1000 population and select the best result from that generation. Then it insert the result of the genetic operations into new generation until reach stopping criteria.

5. Describe your main considerations in splitting the original data set into a training set training.txt and a test set test.txt.

In my code, the rate between training and test data is approximately same with 50%. there are totally 699 instances in breast-cancer-wisconsin. I split 349 instances to test.txt and 350 instances to training.txt. Because there should have larger number of instances in train data. Half of instances is enough for our to train.

6. Report the classification accuracy (average accuracy over 10 independent experiment runs with different random seeds) on both the training set and the test set.

Run 1
(((((((x3 ^ x6) * x4) - x7) - x3) - x3) * x2) - x3) - x5) â€“ x9
Training set classification accuracy: 96.0%
Test set classification accuracy: 95.98853868194843%
run 2
((x2 * ((((x4 * (x3 ^ x6)) - x3) - x3) - x7)) - x7) â€“ x3
Training set classification accuracy: 96.0%

Test set classification accuracy: 95.41547277936962%
run 3
(((((((x3 ^ (x9 * x9)) ^ x6) * x5) - x3) - x7) - x7) - x5) â€“ x1
Training set classification accuracy: 96.28571428571429%
Test set classification accuracy: 94.84240687679083%
run 4
((x2 * ((((x4 * (x3 ^ (x9 * x6))) - x1) - x4) - x1)) - x6) â€“ x7
Training set classification accuracy: 96.57142857142857%
Test set classification accuracy: 95.70200573065902%
run 5
(((((x3 ^ x6) * x5) - (x2 + (x3 ^ x6))) - x3) - x5) â€“ x7
Training set classification accuracy: 96.28571428571429%
Test set classification accuracy: 95.70200573065902%
run 6
(((((x3 ^ ((x5 ^ x6) - x6)) - x5) - x5) - x5) - x5) - x5
Training set classification accuracy: 96.85714285714286%
Test set classification accuracy:
 95.98853868194843%
run 7
(((((((x6 * x2) - (x6 / x1)) ^ x3) - x2) - x2) - x2) - x2) â€“ x2
Training set classification accuracy: 96.28571428571429%
Test set classification accuracy: 93.98280802292264%
run 8
((((((x3 ^ x6) - x6) - (x3 / x5)) ^ x3) ^ x1) - x1) â€“ x7
Training set classification accuracy: 96.57142857142857%
Test set classification accuracy: 95.12893982808023%
run 9
(((((x6 * x2) * x3) - ((x6 / x1) / x1)) - x7) - x3) - x5
Training set classification accuracy: 96.28571428571429%
Test set classification accuracy: 93.69627507163324%
run 10
((((x3 ^ (x9 * (x6 * x4))) - x7) - x7) - x5) - x2
Training set classification accuracy: 96.28571428571429%
Test set classification accuracy: 93.40974212034384%

7. List three best programs evolved by GP and the fitness value of them.

Genotype 1:
(((((x3 ^ ((x5 ^ x6) - x6)) - x5) - x5) - x5) - x5) - x5
Training set classification accuracy: 96.85714285714286%
Test set classification accuracy: 95.98853868194843%

Genotype 2:
((((((x3 ^ x6) - x6) - (x3 / x5)) ^ x3) ^ x1) - x1) â€“ x7
Training set classification accuracy: 96.57142857142857%
Test set classification accuracy: 95.12893982808023%

Genotype 3:
(((((((x3 ^ x6) * x4) - x7) - x3) - x3) * x2) - x3) - x5) â€“ x9
Training set classification accuracy: 96.0%
Test set classification accuracy: 95.98853868194843%

8. (optional, bonus) Analyse one of best programs to reveal why it can solve the problem in the task.

Take 5,1,1,3,2,1,1,1,1=2 and 2,1,1,1,2,1,3,1,1=2

Genotype 1:
(((((1 ^ ((2 ^ 1) - 1)) - 2) - 2) - 2) - 2) â€“ 2 = -9

(((((1 ^ ((2 ^ 1) - 1)) - 2) - 2) - 2) - 2) â€“ 2 = -9

Genotype 2:
((((((1^ 1) - 1) - (1 / 2)) ^ 1) ^ 5) - 5) â€“ 1 = -6.03
((((((2 ^ 1) - 1) - (1 / 2)) ^ 1) ^ 2) - 2) â€“ 3 = -4.75

Genotype 3:
(((((((1 ^ 1) * 3) - 1) - 1) - 1) * 1) - 1) - 2) â€“ 1 = -4
(((((((1 ^ 1) * 1) - 3) - 1) - 1) * 1) - 1) - 2) â€“ 1 = -8