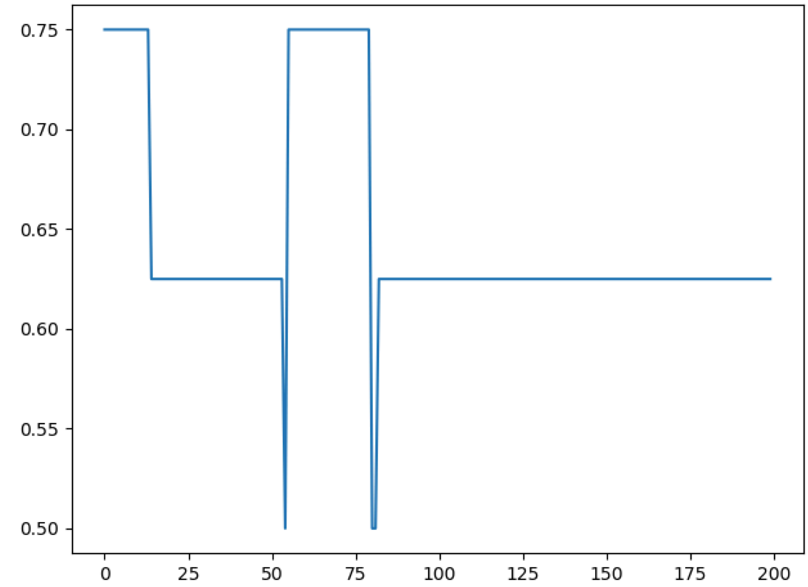
COMP 307 Assignment 2

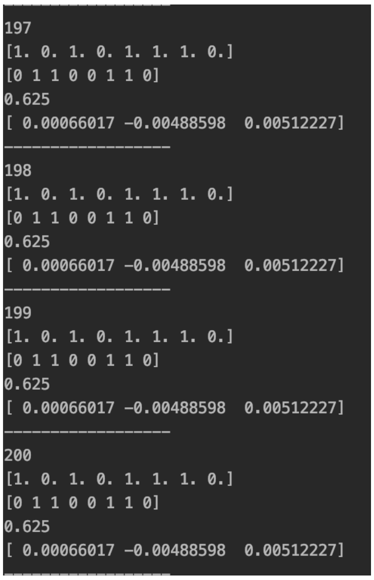
Yan Zichu

300476924

Part 1: Perceptron for Binary Classification

1. Report the classification accuracy your perceptron learning algorithm after running 200 epochs.





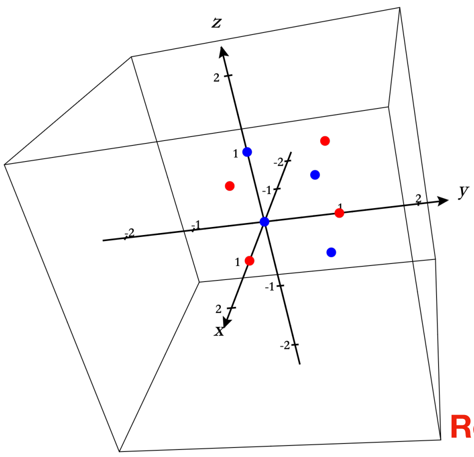
```
197
[1. 0. 1. 0. 1. 1. 1. 0.]
[0 1 1 0 0 1 1 0]
0.625
[ 0.00066017 -0.00488598  0.00512227]
-------------------
198
[1. 0. 1. 0. 1. 1. 1. 0.]
[0 1 1 0 0 1 1 0]
0.625
[ 0.00066017 -0.00488598  0.00512227]
-------------------
199
[1. 0. 1. 0. 1. 1. 1. 0.]
[0 1 1 0 0 1 1 0]
0.625
[ 0.00066017 -0.00488598  0.00512227]
-------------------
200
[1. 0. 1. 0. 1. 1. 1. 0.]
[0 1 1 0 0 1 1 0]
0.625
[ 0.00066017 -0.00488598  0.00512227]
```

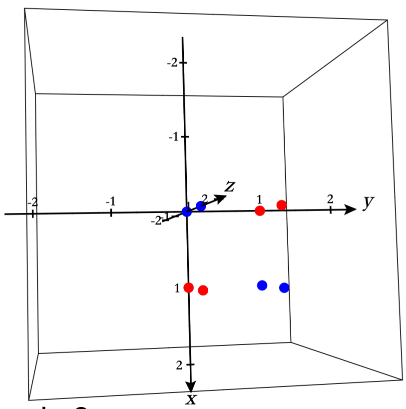
As shown above, the classification accuracy after 200 epochs is 0.625 and the highest performance is 0.75.

We can see that this result is eventually convergent but not good.

2. Analyze and describe reasons that your algorithm could not achieve better results.

The perceptron learning algorithm will converge if and only if the training set is linearly separable. But our data is not linearly separable , shown below:





Red is 1, **Blue** is 0

The points of these two colors cannot be separated by a plane or a curved surface, so actually Perceptron is not very suitable for this dataset. And this is the reason why the accuracy of my model is not high.

Part 2: Neural Networks for Classification (Package : Keras)

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.

Layer	Value	Explanation
Input layer	13	Because our training set has 13 features(columns).
Hidden layer	6	The value should between 13 and 3, and I tired 5 and 7 but they are not work.
Output layer	3	Since our data has three kinds of labels, we will calculate the probabilities of these three labels by NN, and take the label with the highest probability as the predicted value.

If the NN is a classifier, then it also has a single node unless softmax is used in which case the output layer has one node per class label in your model.

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.

When using Adam as optimizer, I ran 2000 epochs first but it was overfitting then I reduced epochs to 1000.

Adam:

Parameter	Value	Explanation
Learning rate	Default(0.001)	When I tuned up learning rate the model converges too fast, which makes it easy to overfitting, but the test accuracy is not high
β1	Default(0.9)	The β1 coefficient is the exponential decay rate, and the control weight distribution (momentum and current gradient) usually takes a value close to 1. I have tried to lower it to 0.1, it will cause insufficient kinetic energy and can not move forward
β2	Default(0.999)	The β2 coefficient is the exponential decay rate and controls the effect of the squared gradient before. Similar to the RMSProp algorithm, the weighted mean of the squared gradients is calculated.

RMSProp:

Parameter	Value	Explanation
Learning rate	0.002	I tuned it from default value 0.001 to 0.003 due to I want it be stable when epochs is 1000. (It is possible to use default value by tuning epoch to 2000 but it is kind of waste memory).

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

termination criteria	Explanation
Accuracy > 0.98	Because when model classification accuracy get 0.98 which means our model actually is fully well-trained and if we keep going is meaningless and will increase the risk of overfitting.
Epochs = 1000	Due to our dataset is small, it is unnecessary to make it large, this number will make sure our learning rate can keep small for converging smoothly.

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

Seed	Training accuracy(4dp)	Test accuracy(4dp)
10	0.9669	0.9342
100	0.9775	0.9662
150	0.9678	0.9543
200	0.9078	0.9053
300	0.9245	0.9345
400	0.9566	0.9432
450	0.9212	0.9056
500	0.9562	0.9552
550	0.9022	0.8923
600	0.8981	0.8899

The overall accuracy is 0.95 on training set and test set, but some of them perform not well I think it was because the epochs is small, the accuracy was still increasing that time. And there are 2 or 3 instances alway predict wrongly so I think there are some outliers in the dataset. In conclusion, neural network is a good way to save this task.

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

The nearest neighbor classification accuracy(97%) is almost same as neural networks(95%) .

KNN does not have training stage and it fully depend on the dataset(features and labels) and k value, so for some small dataset just like this task, it is suitable.

NN require more parameter and it is limited by epoch sometimes because we want to avoid overfitting so tuning down the epoch is the way we chose (and usually we can add Dropout(),but in this task the dataset is too small if we do it it will not get high accuracy).

Overall, for this task I think these two algorithms are both fine.

Part 3: Genetic Programming for Symbolic Regression

1. Determine a good terminal set for this task.

The terminal set contains a constant number between -1.0d and 1.0d and variable X (input value).

2. Determine a good function set for this task.

The function set contains Multiply, Divide, Subtract, Add these basic four arithmetic operator. Sine and Cosine was not added as they sometimes produce 0 which may cause error -denominator is zero.

3. Construct a good fitness function and describe it using plain language, and mathematical formula (or any other format that can describe the fitness function as accurately as mathematical formula, such as pseudo code).

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

Root mean squared error (RMSE): It's the square root of the average of squared differences between prediction(Output value) and actual observation(Given Y value in dataset). And metrics can range from 0 to ∞ and is indifferent to the direction of errors. It is negatively-oriented scores, which means lower values are better.

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

Relevant parameter : Max initial depth = 4
Population = 300
Generation = 100
Max tree depth = 8
Crossover probability = 0.8
Mutation probability = 0.1
Reproduction probability = 0.05

Stopping criteria : Either 100 generation or overall fitness function equals zero. (RMSE =0 is perfect)

5. Report the mean squared error for each of 10 independent runs with different random seeds, and their average value.

Rndom seed	MSE
10	0.000414399999999985
100	0.000414399999999991
120	0.000414399999999993
150	0.000414399999999992
200	0.000414399999999994
250	0.000414399999999983
300	0.000414399999999997
350	0.000414399999999981
400	0.000414400000000001
500	0.000414399999999985

6. List three different best programs and their fitness values.

Case 1:
mul(add(1, add(1, x)), add(1, add(1, add(mul(protectedDiv(0, x), mul(protectedDiv(1, protectedDiv(0, 0)), x)), x))))
Fitness value : 0.00041439999999985
Case 2:
add(mul(x, x), add(add(add(add(x, add(x, add(add(1, x), 1))), protectedDiv(protectedDiv(1, sub(x, x)), 0)), x), protectedDiv(0, protectedDiv(0, 1))))
Fitness value : 0.00041440000000000097
Case 3:
add(x, add(x, sub(sub(x, -1), sub(-1, add(add(add(mul(x, x), x), 1), 1)))))
Fitness value : 0.0004143999999999809

7. (optional, bonus, 5 marks) Analyse one of the best programs and explain why it can solve the problem in the task.

Case 3 :
add(x, add(x, sub(sub(x, -1), sub(-1, add(add(add(mul(x, x), x), 1), 1)))))
Transfer to : Y = X^2 + 3X + 5
When X = 5, Y = 50 , actual Y = 49
When X = 2 , Y = 15 , actual Y = 16
As we can see , there still have error but close to the actual value.

Part 4: Genetic Programming for Classification

1. Determine a good terminal set for this task.
The terminal set contains a constant number between -1.0d and 1.0d and variable V1-V36 (36 variables).

2. Determine a good function set for this task.
The function set contains Multiply, Divide, Subtract, Add these basic four arithmetic operator. Sine and Cosine was not added as they sometimes produce 0 which may cause error -denominator is zero.

3. Construct a good fitness function and describe it using plain language, and mathematical formula (or any other format that can describe the fitness function as accurately as mathematical formula, such as pseudo code).
A good fitness function is a low error rate or high accuracy. I chose high accuracy if ProgOut < 0 then class1 else class2. And when we get correct label ,then plus one on the variable 'count' which is for counting the correct number of prediction. And finally we divide length of prediction by correct number so we can get the accuracy, then GP will select a solution with highest accuracy as the best one, for this task, if ProgOut < 0 then it is 'Anomaly' otherwise it is 'Normal'.

```
def _accuracy(y, y_pred, w):
    """Calculate the accuracy."""
    if y_pred < 0:
        y_pred = 0
    else : y_pred = 1

    diffs = np.abs(y - y_pred) # calculate how many different values

    return 1 - (np.sum(diffs) / len(y_pred))

accuracy = make_fitness(_accuracy, greater_is_better = True)
```

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

Relevant parameter : Max initial depth = 4
Population = 1000
Generation = 2000
Max tree depth = 8
Crossover probability = 0.8
Mutation probability = 0.1
Reproduction probability = 0.05

Stopping criteria : Either 2000 generation or overall fitness function equals 100% accuracy.

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

```
X_train, X_test, y_train, y_test = train_test_split(features, labels, test_size=0.2)
```

Because the original dataset is not large so I need to make sure model training stage has enough data to use, so I left 80% for training set and grab 20% data for test set which shown in the code above.

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.

Rndom seed	Training set accuracy	Test set accuracy
10	98.08%	94.87%
100	98.08%	92.31%
120	100%	100%
150	97.44%	94.87%
200	97.44%	97.44%
250	83.97%	92.31%
300	98.72%	92.31%
350	98.08%	94.87%
400	87.18%	87.18%
500	99.36%	82.05%

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

I selected 3 best from the table above.

100%:

sub(sub(sub(mul(mul(div(V18, V9), sub(mul(mul(div(V6, V1), mul(div(V6, V1), mul(mul(div(V22, V24), mul(div(V18, V9), div(V2, V9))), mul(sub(sub(sub(mul(mul(div(V18, V9), sub(sub(mul(mul(div(V18, V9), sub(mul(mul(div(V6, V1), mul(div(V6, V1), mul(mul(div(V22, V24), mul(div(V18, V9), div(V2, V9))), mul(sub(mul(div(V27, V16), div(V27, V16)), div(sub(V4, V14), V14)), mul(div(V10, V9), div(V22, V24)))))), div(V27, V16)), div(V15, V30))), mul(div(V27, V16), div(V10, V9))), div(V8, V29)), div(V15, V30))), mul(div(V22, V8), div(V10, V9))), div(V8, V29)), div(V8, V29)), div(sub(V4, V14), V14)), mul(div(V10, V9), div(V22, V24)))))), div(V27, V16)), div(V15, V30))), div(V18, V9)), div(V15, V30)), div(V8, V29)), div(V7, V29))

97.44%:

sub(sub(sub(sub(div(add(add(V27, V18), V18), div(mul(div(mul(div(mul(sub(mul(div(add(V9, V5), V18), V20), V18), V8), V10), V9), V2), V5), V22)), div(V19, V6)), div(add(V9, V5), V18)), div(V23, V6)), div(add(V9, V19), V17))

94.87%:

sub(sub(mul(mul(sub(V6, V16), div(add(V36, V31), V10)), div(add(V31, add(V36, V31)), mul(V10, V10))), div(mul(add(V11, add(V13, V7)), V9), mul(V18, V10))), div(div(mul(V30, add(V31, V27)), div(mul(V5, V9), mul(V26, V30))), sub(mul(sub(V6, V32), add(V31, V30)), mul(V35, V13))))

8. (optional, bonus, 5 marks) Analyze one of best programs to identify patterns you can find in the evolved/learned program and why it can solve the problem well (or badly).

97.44%:

sub(sub(sub(sub(div(add(add(V27, V18), V18), div(mul(div(mul(div(mul(sub(mul(div(add(V9, V5), V18), V20), V18), V8), V10), V9), V2), V5), V22)), div(V19, V6)), div(add(V9, V5), V18)), div(V23, V6)), div(add(V9, V19), V17))

Transfer to :

$$y=\frac{V27+2V18}{\dfrac{\dfrac{(V9+V5)V20}{V18}V8V9V2V5}{V10V22}} - \frac{V19}{V6} - \frac{V9+V5}{V18} - \frac{V23}{V6} - \frac{V9+V19}{V17}$$

V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20 V21 V22 V23 V24 V25 V26 V27 V28 V29 V30 V31 V32 V33 V34 V35 V36 Target

41 29 113 129 44 29 113 129 48 37 109 112 46 27 108 129 43 29 108 129 46 32 108 122 52 40 97 105 52 48 90 98 59 63 90 75 Anomaly

By calculate, I got y = -19.2359(4dp) which is lower than 0, so it is belong to Anomaly. And compare with the actual value is correct.

75 91 96 71 79 87 93 71 79 87 93 67 74 87 88 70 78 87 84 70 74 87 88 66 79 88 93 68 79 91 93 72 75 91 93 68 ‘Normal'

By calculate, I got y = 5.9957(4dp) which is bigger than 0, so it is belong to Normal. And compare with the actual value is correct.