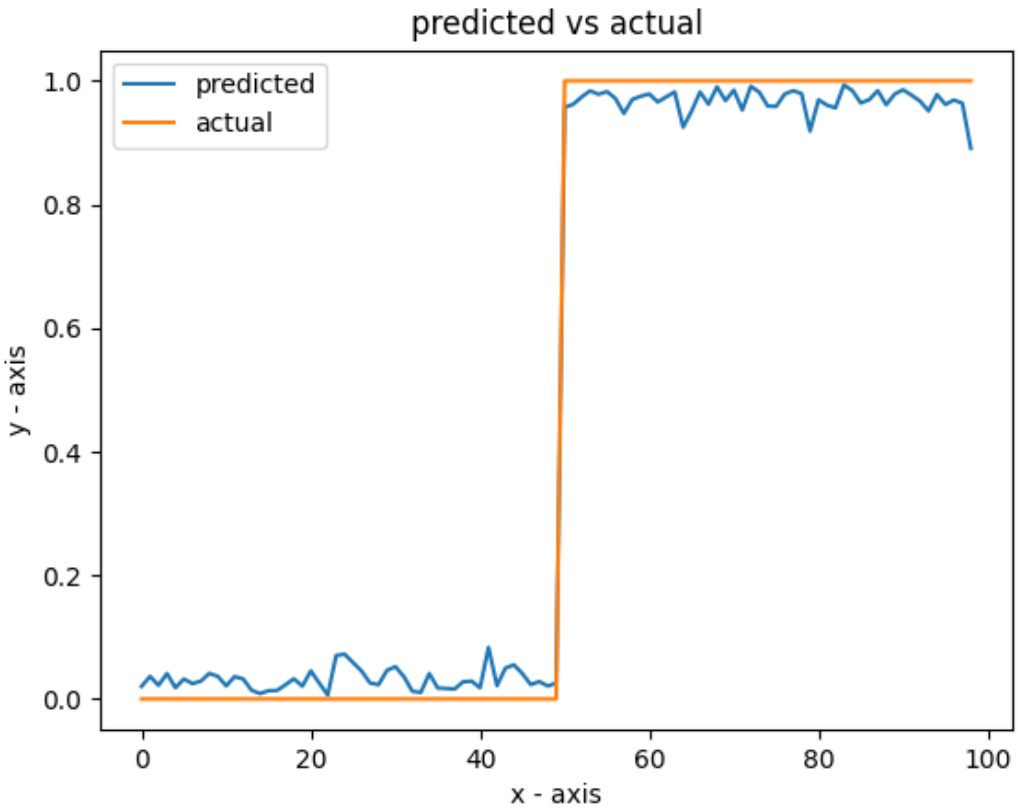
WS4 T1. • You should know which ML approach is more suitable for this kind of binary problem?
• You are asked to use GD. This is like our previous WS, only one thing is significantly different i.e. the use of sigmoid function
• Explain the pre-processing

```
# pre-processing: L2 normalization normalizes values between
0.0-1.0
# creation of b vector, adds bias to the theta_x
```

• The use of sigmoidal function, what is it, and why is it used?
It projects values into a sigmoid curve between 0.0-1.0. It is used to convert regression into classification by encapsulating the cost function and the linear equations into a sigmoid. This way we get the classification probabilities.

• How the GD is operating
Iteratively updates weights in order to minimise the cost function. Gradients are used to choose downhill steps on cost function curves.
• Details on output probabilities and thresholding
The output probabilities are generated through gradient descent. In principal the higher the probability, the higher the chance that a row that we pass through the classifier belongs to a given class. In our case we have set the classification threshold to be 0.5, but other custom thresholds could be explored.

• Comparison between the observed and predicted values, plot them

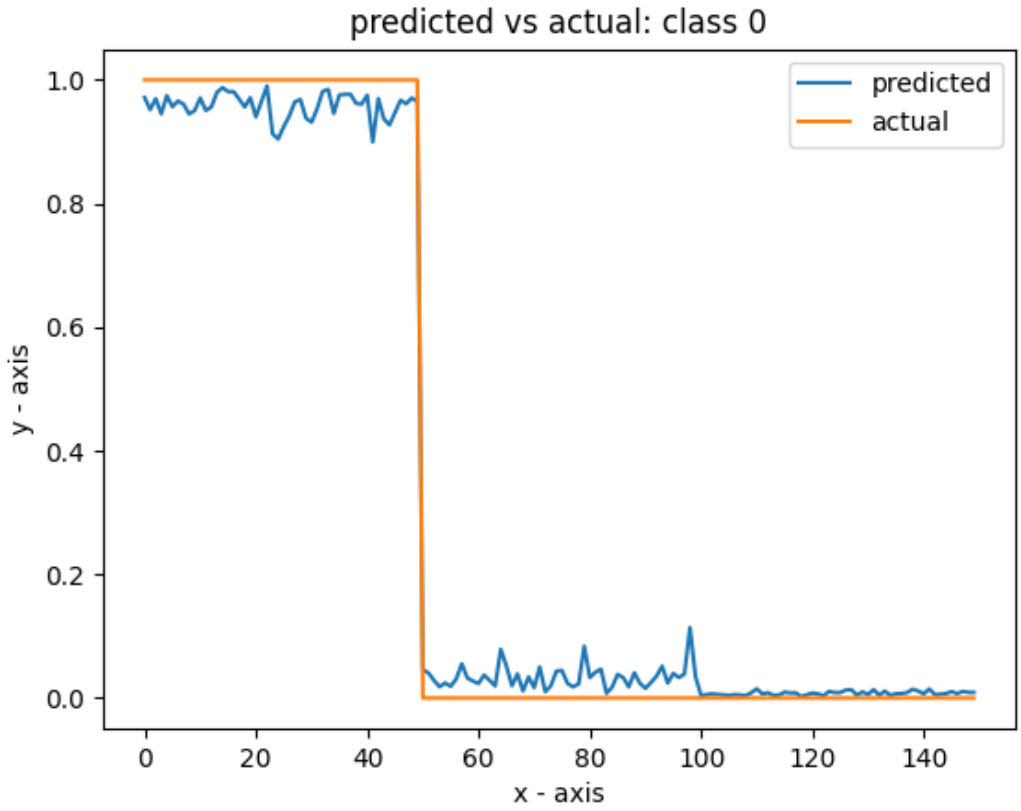


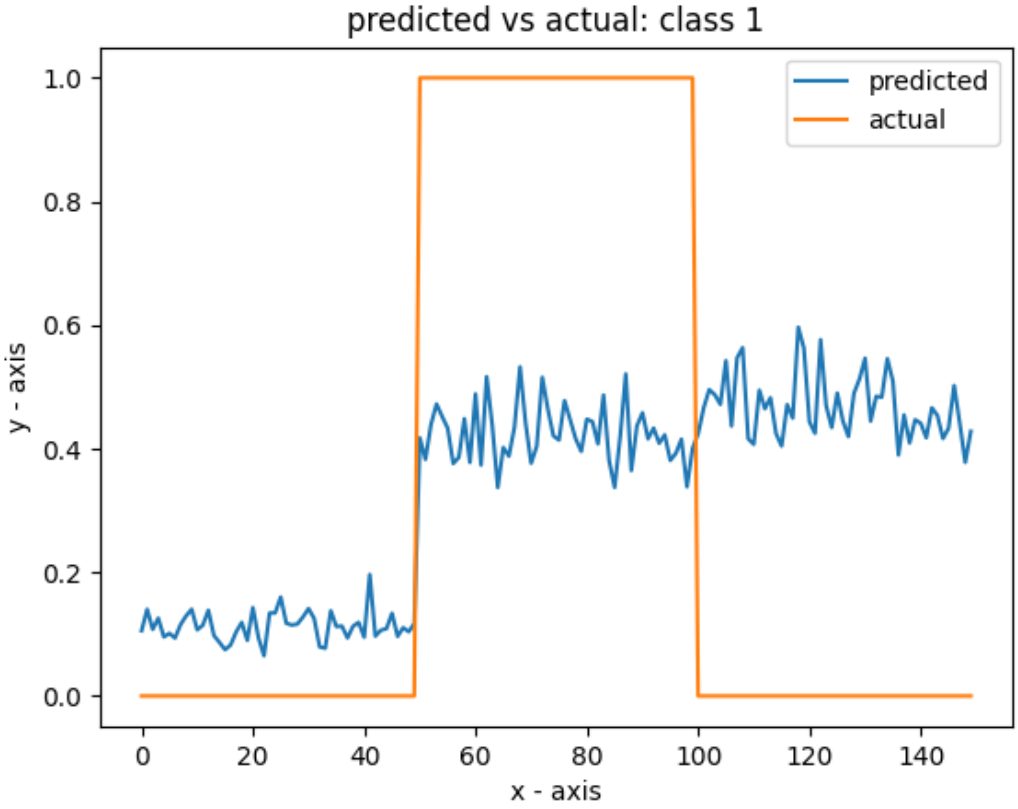
You are given with the full working code, only a few lines are left for you to complete e.g.,: •

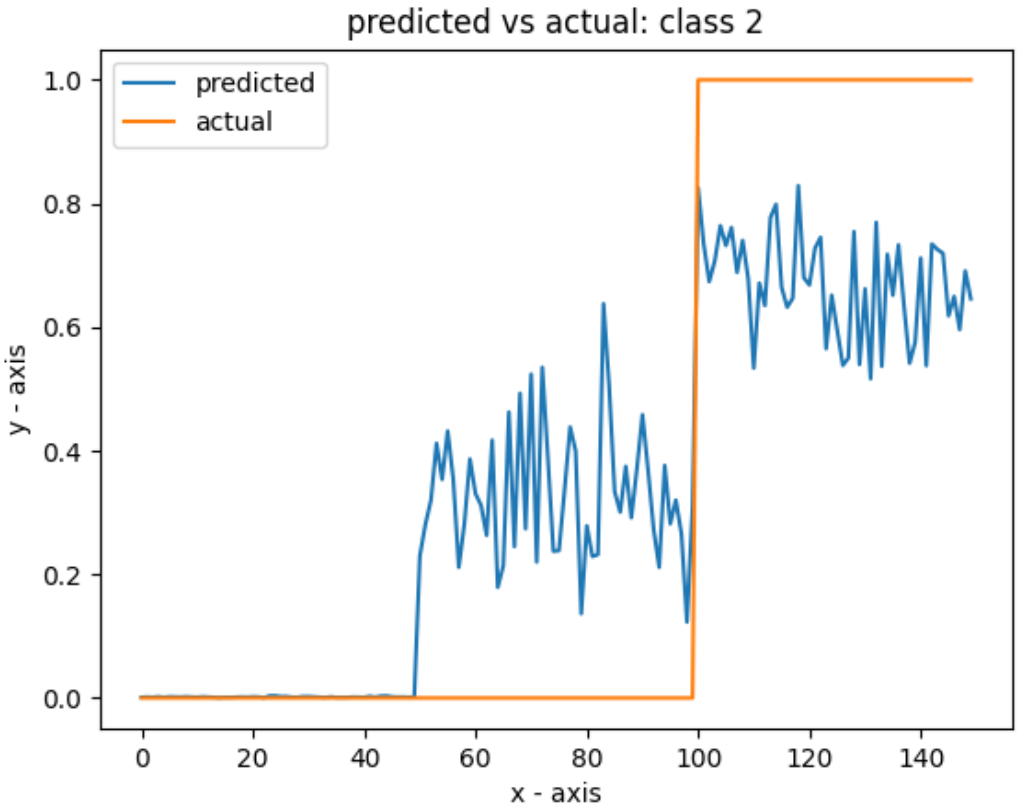
Err= • gradients = • theta = • y_predict = • Code with proper explanation e.g.,
o Explain what the code is doing/ code explanation o Example execution
o Draw results, comparison, and plots in the report.

T2. • You should know which ML approach is more suitable for this kind of multiclassification problem? Softmax
• You are asked to use GD. This is like T1, only one thing is significantly different i.e. softmax
• Explain the pre-processing, including raw data (dataset), one-hot-encoding, why the preprocessing is different from T1

• Explain the use of Softmax function, what is it, and why is it used? For classification. Suppresses probability values between 0,1
• How the GD is operating here
• Details on output probabilities and thresholding (different from T1)
• Comparison between the observed and predicted values, plot them
• Code with proper explanation e.g., o Explain what the code is doing/ code explanation o Example execution o Draw results, comparison, and plots in the report.




predicted vs actual: class 1
predicted
actual
y - axis
x - axis
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100
120
140


predicted vs actual: class 2
predicted
actual
y - axis
x - axis
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100
120
140

WS5 T1. Use BGD and complete the model 1.2. Explain in detail (step by step) the whole process in your own words, including:
• raw data → explain the dataset, show your understanding
• pre-processing (both X and Y (one-hot-encoding)), and other transformations You are given with the full working code, only a few lines are left for you to complete e.g.,: • Err= • gradients = • theta = • y_predict = -
Explain number of iterations/ epochs each epoch is a gradient descent step. At each epoch weights are improving.
- Calculation of above 4 lines
- Plot observed vs predicted values → training results
- Testing the mode on the test data (which is different from the training data) - Plot observed vs predicted values → testing results T2. Use SGD and complete the model 2.2. Explain in detail (step by step) the whole process in your own words, including: • raw data and pre-processing (in the light of SGD/ mini-batches concept) • Number of iterations/epochs (in the light of SGD/ mini-batches concept) • Calculation of above 4 lines • difference between task 1 and 2, in terms of accuracy, explain why they are different e.g., BGD vs. SGD? • Plot observed vs predicted values → training results • Test the model on the test data (which is different from training data) • Plot observed vs predicted results → testing results • Compare the difference b/w training and testing results. WS6 T1. 1.1. Explain in detail (step by step) the forward and backpropagation algorithm with equations, e.g.,: • Starting from a single neuron (linear or logistic regression) • adding more neurons to develop an MLP/ ANN • introduction of new parameters (e.g., a & z), their formulas with explanation • forward propagation and calculation of different parameters, explain this procedure • backpropagation with introduction to error function, weight and bias update equations + explanation 1.2. Complete the equations and run the code • Plot observed vs predicted values → training results 1.3. Test the model on test data • Plot observed vs predicted values → testing results 1.4. Compare results with WS5 Task 2 and comment on results in detail (plot the difference b/w WS5 T2 and WS-6 T1)