**INTRODUCTION MACHINE LEARNING**

**EXERCISE 4**

A red sign with white text

Description automatically generated

|  |
| --- |
| **TEACHER:** |
| Johannes Kiesel |

|  |
| --- |
| **GROUP:** |
| Group 16 |
|  |
| **SUBMITTED BY:** |
| Aaron Perez Herrera |
| Cesar Fernando Gamba Tiusaba |
| Chun Ting Lin |
| Olubunmi Emmanuel Ogunleye |

Exercise 1 : Gradient Descent (1+1+1=3 Points)

(a) Name one difference between the perceptron training rule and the gradient descent method.

The Perceptron training rule is not based on residuals in the (p+1) dimensional input- output-space) but refers to the input space only, where it simply evaluates the side of the hyperplane as a binary feature (correct side or not). Gradient descent is a regression approach that exploits the residuals provided by a loss function of choice, whose differential is evaluated to guide hyperplane search

(b) What are the main requirements for the hypothesis space and error function for gradient descent to be successfully applied?

In batch gradient descent, the weight vector is determined by calculating the differences in weights for each example in the training data and subsequently computing the weight vector for the entire batch in a single step. Incremental gradient descent involves computing the weight vector immediately after calculating the difference in weights for each example in the training dataset, leading to a more iterative and continuous update process. Compared to batch gradient descent, the example-based weight adaptation of incremental gradient descent can better avoid getting stuck in a local minimum of the loss function.

(c) Name the key difference between the algorithm of batch gradient descent (BGD) and the algorithm of incremental gradient descent (IGD).

The perceptron y1(), with a non-zero bias term wo. xo, is more general than yo(), which has a bias term of zero. The bias term acts as an additional flexibility factor, allowing y1() to represent all the functions yo()'s weights can capture, plus additional functions that arise from the non-zero bias. In essence, y1() can model a broader range of relationships in the data, making it more versatile and general than yo(). The bias term allows models to fit the data better by providing additional flexibility.

Exercise 2 : Perceptron Learning (2 Points)

Consider two perceptrons, yo () and y1 (), both defined by the function heaviside(j\_owjj). Both perceptrons have identical weights except for the bias term: wo = 0 for yo() and wo = 1 for y1(). Determine if one of the perceptrons, yo() or y1 (), is more general than the other (as defined in the lecture units on concept learning). If one is more general, specify which one and explain your answer.

Task in progress (cesar)

Exercise 3 : Perceptron Learning (1+1+2+2+1=6 Points)

In this exercise, you design a single perceptron with two inputs a1 and x2. This perceptron shall implement the boolean formula A A -B with a suitable function y(x1, x2). Use the values 0 for false and 1 for true.

(a) Draw all possible examples and a suitable decision boundary in a coordinate system.

(b) Draw the graph of the perceptron. The schematic must include a1, x2, and all model weights.

(c) Manually determine the weights w = (wo, w1, w2) for the decision boundary you drew in (a).

(d) Now determine w using the perceptron training algorithm (PT). Use a learning rate n of 0.3 and initialize the weights with wo = -0.5 and w1 = w2 = 0.5. Instead of selecting examples randomly, use the following examples in the given order (stop after those four examples):

|  |  |  |
| --- | --- | --- |
| X1 | X2 | C |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Draw the decision boundary after every weight update into the coordinate system of (a).

(e) Briefly describe one effect of changing the learning rate n on the learning progress.

Task in progress (cesar)