Đề tham khảo 2

▼ Source

Artificial Intelligent and Machine Learning

Time: 1h30

Students are allowed to use their personal calculator

Exercise 1 (5points)

Explain how the two methods for linear regression problem:

- Gradient descent
- Normal equation

Exercise 2 (5 points)

- Given the training data for a linear regression problem as follow:

Input	Output
0	0
1	2
-1	-2
2	3

- After the first iteration, the values of two coefficients are:
- Θ_0=2 and Θ_1=1
- What are the initial values of Θ_0 and Θ_1? Given the learning rate α equals to 4.

Exercise 3 (5 points)

Give the following confusion matrix. Assume that the accuracy of the classifier is 80%. Calculate x,y and other scores precision, recall, specificity and f-score.

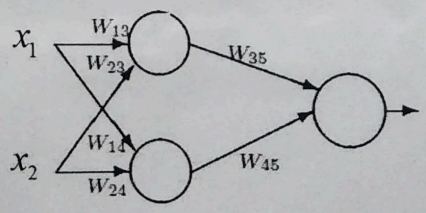
	Predicted: 0	Predicted: 1
Actual:0	800	y
Actual:1	x	900

Exercise 4 (5 points)

Calculate the output of the following neural network with two inputs x1=1 and x2= -1, the activation function is described as follow:

$F(x)= x^2 + 2x + 3$

$w_{13} = 2$ $w_{23} = -3$	$w_{35} = 2$
$w_{14} = 1$ $w_{24} = 4$	$w_{45} = -1$



Exercise 1

Explain how the two methods for linear regression problem:

- Gradient descent
- Normal equation

✅ **1. Gradient Descent:**

Gradient Descent is an iterative optimization algorithm used to find the minimum of a function. Here's how it works in the context of linear regression:

- We start with an initial guess for the parameters (weights) of our model.
- We calculate the cost function, which measures how well our model fits the data. For linear regression, this is usually the Mean Squared Error (MSE).
- We compute the gradient of the cost function at our current point in the parameter space. This gives us the direction of steepest ascent.
- We update our parameters by taking a step in the direction of steepest descent (opposite to the gradient).
- We repeat these steps until we reach a point where the cost function is minimized.

The learning rate determines how big of a step we take on each iteration. If it's too small, the algorithm will be slow to converge. If it's too large, we might overshoot the minimum and cause the algorithm to diverge.

2. Normal Equation:

The Normal Equation is a mathematical approach that provides an analytical solution to the linear regression problem. Unlike Gradient Descent, which is an iterative method, the Normal Equation solves for the parameters in just one step.

Here's how it works:

- We set up a system of linear equations based on our dataset (one equation for each data point).
- We solve this system using matrix operations to find the parameters that minimize our cost function.

The advantage of this method is that it doesn't require any feature scaling, and you don't need to choose a learning rate like you do with Gradient Descent. However, it can be slow if your dataset is very large, because computing the inverse of a matrix is a costly operation.

In summary, Gradient Descent and Normal Equation are two different approaches to solve linear regression problems. Gradient Descent is generally preferred when dealing with large datasets due to its efficiency and scalability, while Normal Equation can be faster and more direct for smaller datasets.

Exercise 2

- Given the training data for a linear regression problem as follow:

Input	Output
0	0
1	2
-1	-2
2	3

- After the first iteration, the values of 2 coefficients are:
- $\theta_0 = 2$ and $\theta_1 = 1$
- What are the initial values of θ_0 and θ_1? Given the learning rate $\alpha = 4$

In a linear regression problem, the parameters θ_0 and θ_1 are updated using the gradient descent algorithm. The update equations are as follows:

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$$

where:

- m is the number of training examples,
- $h_\theta(x)$ is the hypothesis function, defined as $h_\theta(x) = \theta_0 + \theta_1 x$,
- $x^{(i)}$ is the input of the i-th training example,
- $y^{(i)}$ is the output of the i-th training example,
- α is the learning rate.

Given that after the first iteration, $\theta_0 = 2$ and $\theta_1 = 1$, and the learning rate $\alpha = 4$, you can substitute these values into the update equations and solve for the initial values of θ_0 and θ_1. Note that you'll need to calculate $h_\theta(x)$ for each training example in each sum.

Exercise 3

Given the following confusion matrix. Assume that the accuracy of the classifier is 80%. Calculate x, y and other scores precision, recall, specify and f-score.

	Predicted: 0	Predicted: 1
Actual: 0	800	y
Actual: 1	x	900

⚠️ The accuracy of a classifier is calculated as the sum of true positives and true negatives divided by the total number of examples. In this case, the accuracy is given as 80%.

From the confusion matrix, we know that there are 800 true negatives (Actual: 0, Predicted: 0) and 900 true positives (Actual: 1, Predicted: 1). Let's denote the total number of examples as N. We can write the accuracy as:

$$\frac{800 + 900}{N} = 0.8$$

Solving this equation gives $N = 2125$.

The sum of each row in the confusion matrix gives the actual number of examples in each class. Therefore, we have:

- For Actual: 0, $800 + y = N/2 = 1062.5$
- For Actual: 1, $x + 900 = N/2 = 1062.5$

Solving these equations gives $x = 162.5$ and $y = 262.5$.

Now we can calculate precision, recall, specificity, and F-score:

- Precision (for class 1) is the number of true positives divided by the sum of true positives and false positives: $\frac{900}{900+y}$
- Recall (for class 1) is the number of true positives divided by the sum of true positives and false negatives: $\frac{900}{900+x}$
- Specificity (for class 0) is the number of true negatives divided by the sum of true negatives and false positives: $\frac{800}{800+x}$
- F-score is the harmonic mean of precision and recall: $2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$

Substitute x and y into these formulas to get the final results.