7PAM2021-0105-2024-Machine Learning Tutorial

NAME: DANUSHMATHI PATHMANABAN STUDENT ID: 23072008

Github link: https://github.com/DANUSHMATHI2002/ML_Assignment.git

---

MACHINE LEARNING TUTORIAL TRANSCRIPT

00:00:02 Introduction

hi today i will be presenting on multi-layer perceptrons commonly known as mlps this tutorial will cover the structure working mechanism and how we can implement them in python using the wine quality data set let's get started these are the content that i am going to discuss in this tutorial today introduction to mlps structure of an mlp, activation functions, training and mlp and data set for mlp training ,coding and mlp in python

00:00:32 What is an MLP?

First, let's see about the introduction to MLP. Let's see what is an MLP and let's get started with the definition of an MLP. Artificial neural network or fundamental part of deep learning and one of the most commonly used architectures is the multi-layer perceptron. An MLP consists of multiple layers of neurons that help process and learn from the data.

00:00:58 Applications of MLPs

is fully connected meaning every neuron in one layer is connected to every neuron in the next layer this allows mlps to learn complex patterns in the data next let's see the classification and regression applications used in the mlps mlps are commonly used for tasks such as classification where we can identify the categories or label example in the spam detection in regression where we can predict the continuous output next

00:01:24 Goal of this Tutorial

The goal of the tutorial is to demonstrate the workings of MLPs and guide you in building a simple MLP using the Python and assist you in analyzing a real dataset.

00:01:39 Visual Overview of an MLP

let's see the visual overview of MLP the structure of an MLP includes three components an input layer one or more hidden layers and an output layer there is an input layer which consists of an input which is given here so each layer contains some neuron which applies a mathematical transformation to the inputs which we give here the connection between the neurons so the connections between the neurons

00:02:02 Key components of an MLP

haven't weights that are adjusted during the training to improve the prediction. This diagram helps clarify how MLP architectures or structure. So there are key components in which we came to know that there is a

neurons in each layer and there is a weight, which means the connections and the activation functions. So the activation functions are one day in the hidden layers that says how the data is transformed through successive layers.

00:02:27 Structure of an MLP

so in this activation functions it says how the data is transformed through the successive layer to get an output layer and next let's see the structure of an mlp so let's see how does an mlp work before we saw the structure of an mlp now let's see what what does the each layer do to.

00:02:52 Mathematical Foundation of MLPs

Let's see the mathematical foundation as in the before slide we have seen the mathematical formula. Now let's see what the each term represent. So the W represent the weight matrix that represent the connection trends between neurons across each layers. So by adjusting these weights during the training improves the model's ability to make the predictions accurate.

00:03:15 Impact of biases

Next let's see about the impact of biases. So the biases term is added to the weighted sum of input to allow the model to fit the data better. So it acts as an offset that can shift the activation function enhancing the model performance. Next let's see the role of activation function. So what is the main role of an activation function is that it introduces the non-linearity into the model.

00:03:39 Activation Functions

To learn and approximate the complex relationship between the data. So without activation functions, MLPs would behave like a linear model. So it is an important to have an activation function in each MLPs. So now let's see why it is important to have an activation functions in each MLPs. Let's see the importance of activation function, why activation function matters.

00:04:06 Importance of Activation Functions

So without activation function, neural networks would behave like a normal linear regression models and fail to capture the complex relationships. Activation function help network learn from nonlinear data and make accurate predictions. So activation functions are crucial because they allow the neural network to capture the nonlinear relationships in the data. So it is suitable for a range of predictive tasks that linear models cannot handle.

00:04:35 Types of Activation Functions

So the common type of activation function are ReLU, TanH and Sigmoid. So each function have a different properties and effects on model training and performance. Examples and applications for instance ReLU is widely used in hidden layers. So this is the ReLU which is used in the hidden layers which cannot be seen from the outside to effectively combat the vanishing gradient problem.

00:04:56 How the functions are used for different purposes?

Which is used for the gradient problem. In contrast, sigmoid functions can be applied in output layer for binary classification tasks. So, the sigmoid function for the output layer and in the hidden layer, we use the ReLU layer. We can see about them in detail in the upcoming slides. Let's see the types of activation functions and look at the three commonly used activation functions. The first is the ReLU and the second is the sigmoid and third is the tanh.

00:05:21 Uses of Activation Functions

Relu, which is fast and efficient for deep networks, works well in hidden layers. And second is the SIGMOID, which is useful for probability, best for binary classification, but suffers from vanishing gradients. And third is the TANH, which is useful for zero-centered values, similar to SIGMOID, improved optimization. Next, let's see about the training and MLP. So, there are two processes. Let's see them.

00:05:47 Forward Propagation

Let's see the learning process of an MLP. First, let's go through the forward propagation explanation. Data moves from the input to the output making the predictions. In forward propagation, the input passes through multiple layers. Each neuron processes the data using the weights, biases and activation function to generate the predictions at the output layer. Next, let's see about the back propagation mechanism. In back propagation, the model adjusts weights using errors to improve the accuracy.

00:06:17 Backward Propagation

Backward propagation is the learning process where errors are calculated and used to update the width. It follows the steps, compute the prediction error, use gradient descent to minimize the error, adjust weight and biases accordingly. This process ensures that the model improves with each iteration. This cycle continues until the model achieves the optimal performance.

00:06:45 Mathematical Concepts in Training

Let's see the mathematical concepts in training. So the first is the loss function which is the mean squared error and the second is the weight update rule via the gradient descent. The mean squared error is a common loss function used to measure the average square difference between the predictor and the target values. So training an MLP involves optimizing a loss function using techniques like gradient descent. The objective is to minimize the errors by adjusting the weights through back propagation.

00:07:15 Exploration of Dataset

So now it's the time we came to know about how the structure of an MLP looks like and how does the each layer of an MLP works and what are the layers in it, input layer, output layer and the hidden layer and how does the forward propagation works, how does the backward propagation works and what does the activation function matters, what is bias and weight. Now it's the time to explore the real life example to get clear about the idea. So let's explore the dataset for MLP training.

00:07:38 Exploring the Wine Quality Dataset

For our practical implementation, we will use the wine quality dataset from the UCL machine learning repository. This dataset contains chemical properties of wines and we will train our MLP models to predict their quality ratings, the features and labels in the dataset. The dataset includes the features such as acidity, sugar content, alcohol content. The output labels are wine quality ratings on a numerical scale.

00:08:04 What is the goal of the dataset?

Our goal is to build an MLP model that can predict these ratings accurately. Next, let's see the coding of an MLP in Python. Let's see how we can code an MLP in Python. Now let's move to the coding part. We will load the dataset.

00:08:21 Implementing an MLP in Python

Pre-process the data, define the MLP architecture, train and evaluate the model. We will use the TensorFlow and Keras for implementation. So the model architecture definition, we will involving the specifying the number of layers, number of neurons in each layer and activation functions. The steps to load and pre-process the data followed by pre-processing which includes normalizing the features. Training and evaluating the model is conducted using the training dataset.

00:08:46 Example

Post-training and the post-testing is involved in getting the accuracy of the model. Let's get example. Let's first see the example with the theoretical problem and next let's check the output with the coding part. So we have a basic MLP with two input neurons. So the one hidden layer with two neurons and one output neuron. So the radio activation function is taken here. So the inputs are as follows. The X1 is equal to 0.6 and the X2 is equal to 0.8. So it is the acidity content and this is the alcohol content of the wine.

00:09:12 Example Calculation of Forward Propagation

So here we take the initial weights and biases which is randomly initialized. Let's see the theoretical example of the path and let's check the answer with the theoretical path with the solution in the code if it matches the answer which we got is correct. So first let's take the input as the x1 x2 weights and bias and activation functions as follow.

00:09:33 Example Calculation of Activation Function

which is chosen at random. First step is to compute the weighted sum. The computed weighted sum is calculated by taking the first input value into the first weight and the second input value in the second weight plus the bias. The bias is fixed for both input and the weight. And next, let's calculate the activation function for this given value. So when we are calculating with this formula, we get the 0.48 for the computed weighted sum. With this computed weighted sum, we need to calculate the activation function.

00:10:01 MSE(Mean Square Error)

We need to replace the 0.48 as x in the activation function so we get the output from this neuron is 0.48 so with this we need to calculate the loss calculation so the predicted output is 0.48 as we got from here and the actual output is 1.08 to compute the loss we use the mean squared error

00:10:17 Loss Calculation Example

So the formula is L is equal to 1 by 2 y minus y by the whole square. So in this y is 1.0, which is the actual output minus the predicted output. So we need to calculate the L value. So the mean square error we get is 0.1352. So the loss is 0.1352. So as before I have calculated for the forward propagation. If it is needed, we can check the backward propagation in the coding part.

00:10:43 Backpropagation and Weight Updates

Next is the visualizing the model performance to access the model performance we can plot the loss curves to check how training progress evaluation metrics like accuracy or mean of the later these visualization help identify overfitting or underfitting issues now let's see the coding now let's input the essential libraries and let's load the wine quality data set which is taken from the UCL.

00:11:04 Visualizing Model Performance

data repository and we are splitting the data into features and target variable. We are normalizing the features. We are splitting the dataset into training and testing. We are defining the MLB model. You are taking the ReLU activation function and we are compiling the model. Next, we are training the model with the approach is equal to 100, which is of 100 value and verbose is equal to 1. So the approach is taken of 100.

00:11:25 Visualizing Model Performance and MSE

And now let's plot the training history to see the mean absolute error and the model performance over the approach. So we can see that the mean absolute error and the model performance over the approaches aligned together, which is perfectly good. Now we are next, we are going to evaluate the model after evaluating the approach.

00:11:41 Final Model Evaluation

how it performs over the mean obsolete error. So we came to know that the mean obsolete error over here is 0.4964. Now from the PPT, we have verified a theoretical example. We have given an inputs to check whether the model is working correctly and the predictor outcome is correct. So we, first we are going to check with the forward propagation by computing the weighted sum form now for the given input, we have 0.4, right?

00:12:06 Checking the output

So, as i mentioned before so it is 0.48 which is calculated over here is same as we calculated over in the coding path next we are going to evaluate for the loss calculation with using the mean square error value so we got 0.1352

00:12:22 Conclusion

So we got the same as 0.1352 with the same formula as in the coding part as well as the theoretical part and next let's see the back propagation the weight update using the gradient descent we have the 0.1740 taking the gradient value is equal to 0.26 we have run the model successfully and it is working successful thank you for watching the tutorial.