Contents

Chapter 1

Neural Nets for Regression

Neural Networks are powerful computational models that are used for a variety of tasks in machine learning, including regression. Unlike linear regression, which is a linear approach to modelling the relationship between a dependent variable and one or more independent variables, a Neural Network can model complex non-linear relationships.

1.1 Neural Networks

A neural network is composed of nodes (neurons) grouped into layers. The input layer takes in the features of the dataset, hidden layers perform computations on these inputs, and the output layer produces the final prediction. Each node in a layer is connected to each node in the next layer through ”edges”. These edges carry weights, which are the parameters of the model that are learned during training.

A key part of each node is an activation function. It transforms the weighted sum of the node’s inputs into an output value that is passed onto the next layer. Common activation functions include the ReLU (Rectified Linear Unit), sigmoid and hyperbolic tangent functions. For regression tasks, usually a linear activation function is used in the output layer, as the output can be any real number.

1.2 Neural Networks for Regression

To perform regression using a neural network, you would train the network to map the input features to a continuous target variable.

Here is the basic process:

- **Feedforward:** Compute the output of the network given the input features. This involves calculating the weighted sum of inputs for each node and applying the activation function.
- **Loss Calculation:** Calculate the loss (difference between the network's prediction and the actual value). For regression tasks, common loss functions include Mean Squared Error (MSE) or Mean Absolute Error (MAE).
- **Backpropagation:** Update the network's weights to minimize the loss. This is done by computing the gradient of the loss function with respect to each weight in the network, and then adjusting the weights in the direction that decreases the loss.
- **Iteration:** Repeat the feedforward, loss calculation, and backpropagation steps for a number of epochs (complete passes through the dataset) or until the loss converges to a minimum.

This way, the neural network learns to approximate the function that best maps input features to the target variable, thus performing regression. The advantage of using neural networks over linear regression is that they can capture complex non-linear relationships between variables.

CHAPTER 2

Neural Networks for Classification

Neural Networks can also be used for classification tasks, which involve predicting a discrete class label output for an instance. The process of using a Neural Network for classification is similar to using it for regression, but there are key differences in the output layer and loss function.

2.1 Neural Networks for Classification

For a binary classification problem, where the output can be either of two classes, the output layer of the neural network typically consists of a single neuron with a sigmoid activation function, which squashes the output between 0 and 1. This output can be interpreted as the probability that the instance belongs to a particular class.

For a multi-class classification problem, where the output can be one of more than two classes, the output layer typically has as many neurons as there are classes, and a softmax activation function is used, which gives the probability distribution over the classes.

Here is the basic process:

- **Feedforward:** Compute the output of the network given the input features, just as in regression.
- **Loss Calculation:** Calculate the loss (difference between the network's prediction and the actual class). For classification tasks, common loss functions include Cross-Entropy Loss.
- **Backpropagation:** Update the network's weights to minimize the loss, the same way as in regression.
- **Iteration:** Repeat the feedforward, loss calculation, and backpropagation steps for a number of epochs (complete passes through the dataset) or until the loss converges to a minimum.

Once trained, the neural network can classify a new instance by performing a feedforward pass and predicting the class with the highest probability in the output layer.

Chapter 3

Deep Learning

Deep learning is a subfield of machine learning that focuses on algorithms inspired by the structure and function of the brain called artificial neural networks. While you may have encountered simple, shallow neural networks, deep learning involves neural networks with many layers, hence they are often referred to as ”deep” neural networks.

3.1 Deep Learning

Deep learning models learn to represent data by training on a large number of examples. Unlike shallow neural networks that have one or two layers of hidden nodes, deep networks can have tens or even hundreds of layers of hidden nodes. Each layer in these networks performs a nonlinear transformation of its inputs and is trained to extract increasingly abstract features with each additional layer.

3.2 Chain Rule

In order to understand how these networks are trained, we need to revisit a fundamental concept from calculus, the chain rule. The chain rule is used for differentiating compositions of functions. It essentially says that the derivative of a composed function is the product of the derivatives of the composed functions.

Suppose we have a function $y = f(g(x))$, then the derivative of y with respect to x is:

$$\frac{dy}{dx} = f'(g(x)) \cdot g'(x) \tag{3.1}$$

This rule becomes indispensable when calculating the gradient of the loss function in a deep learning model with respect to the model parameters.

3.3 Backpropagation

Backpropagation is the method used to train deep learning models by calculating the gradient of the loss function with respect to each weight in the network. The name "backpropagation" comes from the fact that the calculation of the gradient proceeds backwards through the network, with the gradient of the final layer of weights being calculated first and the gradient of the first layer of weights being calculated last.

Mathematically, backpropagation uses the chain rule to efficiently compute these gradients. Starting from the final layer, the chain rule is repeatedly applied to propagate the gradient backwards through the network, storing intermediate results as it goes along. Once the gradient has been calculated, the weights are updated using a gradient descent step.

Appendix A

Answers to Exercises

Index