Neural Networks

Pranay Patil

1 Introduction

This document introduces neural networks and is intended for readers with no prior knowledge of these types of networks.

You are free to distribute this document as you wish. Please report any mistakes to patilpranay@berkeley.edu.

Contents

2 Notation

Notation	Meaning
$\mathbb{R}$	the set of real numbers

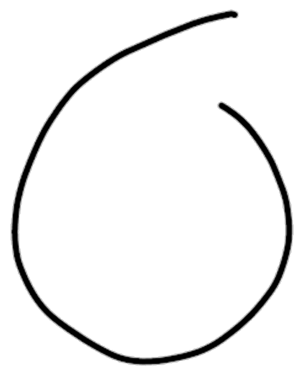
2.1 Other Notes

- Some of the entries in the table shown above may not make complete sense to the reader as of yet. These entries are intended to be a reference for the reader after they have first been introduced to the reader in the following sections of this document.

3 Motivation

Before discussing neural nets, let us take a step back and discuss the broader field of **machine learning**.

There are lots of problems, such as recognizing handwritten digits or identifying spam, for which it is too difficult to write an explicit computer program that always outputs the correct answer. This is primarily due to the variability of the data, as there are no specific rules that dictate whether the following digit is a zero or a six or if an email should be tagged as spam or not.



For this reason, we build models that learn by looking at lots and lots of examples and their correct answers. Then, when the model comes across a particular example that it has not seen before, it uses what it has learned to output a prediction for this new example.

The field of machine learning is concerned with the study of these types of models and what strategies and algorithms to use to build the best model, i.e., one that minimizes the number of incorrect predictions for unseen examples.

3.1 Formulation

We can formally state the problem described above as follows.

Given $\{x_i, y_i\}$ for $i = 1 \dots n$, we want to build a model f such that we minimize $\sum_{i=1}^{n} \|y_i - f(x_i)\|^2$.

Let us break down this statement and understand what it is saying. $\{x_i, y_i\}$ indicates a set of **training data** of size n, where the x_i are the examples and the y_i are the labels (or correct answers) of these examples.

f represents a model (which can be thought of as a function) that, given an input x_i, outputs a prediction $f(x_i)$.

Since we don't know the correct answers to any unseen data, the only way we can judge how well our model works is to compare the predictions output by the model against the actual labels of the training examples. We want the difference between the label and the predicted answer to be as small as possible, i.e., we want to minimize $y_i - f(x_i)$. We take the squared l_2 norm[1] of this value to discard the effect of negatives, resulting in $\|y_i - f(x_i)\|^2$. This as known as the **error** of the model for example x_i.

[1]The l_2 norm of a d-dimensional vector x is $\sqrt{x_1^2 + \dots + x_d^2}$.

Since we want to measure how well the model predicts against all of the training examples, we calculate the error of the model for all the examples and sum up the results, giving us

$$\sum_{i=1}^{n} \|y_i - f(x_i)\|^2.$$

This is known as the **squared error loss** and is one way to measure the quality of a model.

3.2 Features

Now that we know what exactly a machine learning model is and how to quantify its effectiveness, let us take a closer look at the input to such a model, i.e., the training data.

As described above, the training data consists of training examples (the x_i) and their corresponding labels (the y_i). However, while it is simple to numerically represent a label, with $y = 7$ indicating a handwritten '7', it is not as obvious how to do so for the actual training example itself. How do you numerically represent an image of a handwritten digit?

To do so, we must extract **features**, measurable properties of an object, from the image. In this case, a potential choice of features would be the values of each of the pixels of the image.

Once the features are chosen, they are encoded as a d-dimensional vector where d is the number of features. In the case of a 28x28 image, d would be the number of pixels, or 784. Thus, the x_i that we have up until this point used to represent the training examples are actually all d-dimensional vectors where the components of x_i are the values of the features of example i.

3.3 Feature Extraction

One of the biggest challenges to building an effective machine learning model is choosing which features of the training examples to use to train (or build) the model. Just feeding raw data to the model usually does not work and so the performance of many models is dependent on how insightful the programmer is in choosing the features.

The advantage of using neural nets is that they are able to learn by themselves which features to focus on, thus moving the problem of feature extraction from the programmer to the computer.

4 Neural Networks