Perceptron

Sam Gilbert
a1737770

Abstract

The perceptron algorithm is one of the simplest machine learning algorithms. It is a form of supervised learning that

1. Introduction

Perceptron is one of the simplest machine learning algorithms. Perceptron is a supervised learning algorithm that takes a combination of parameters to make a binary classification. Perceptron works by iteratively calculating a set of weights based on the dimension size of the features in the dataset, D. The output of which is a line or D dimension plane that separates the observations into two categories. Perceptron calculates the updated weights iteratively and in an ideal world will converge to a D dimension vector of values that separates all values into their correct groups. Where the data is not able to be separated cleanly into two groups the Perceptron algorithm will never converge. This requires a max iteration value to be set such that the problem is bounded. If a max iteration value is not set the Perceptron algorithm can keep moving back and forth trying to cleanly separate the training observations into the correct groups. Determining the optimal max iterations for the dataset will be investigated in the method section of the report.

In this project the Perceptron algorithm was implemented on the Pima Indians Diabetes Database open source dataset [2]. This dataset contains a set of eight medical predictor variables, the number of pregnancies the patient has had, their BMI, insulin level, plasma glucose concentration, blood pressure, skin fold thickness, age and their diabetes pedigree. The dataset also contains an outcome variable which is either the patient has diabetes or they don't have diabetes. The predictor variables are each one of the D dimensions.

2. Method

As outlined in the introduction section the goal of the Perceptron algorithm is to separate the entries in the dataset into one of two binary groups. The data is read into a python array of dictionaries with the features being stored in a D dimensional array and the outcome stored as a signed integer. The data is then split into a training and testing set using the scikit learn train test function. An 80-20 training testing split was used [1].

The Perceptron algorithm takes an input layer which is the set of features in each observation, let the vector of features for observation i be defined as x_i. Let the collection of these observations be defined as X. Each of these observations have an outcome variable which is whether the patient has diabetes or not. Let the outcome for the i_{th} observation be defined as y_i. This outcome variable is either $+1$ if the patient has diabetes or -1 if the patient doesn't have diabetes. Let the collection of all the outcomes for each observation be defined as Y. The Perceptron algorithm starts by creating a D dimension vector of weights, defined as W. For this implementation the initial W is set as a D dimension zero vector. A bias value is also required which is defined as b initially set to 0. This bias shifts the intercept of the dimensions linear separator away from 0 and is required as it may not be possible to split the data points if this doesn't occur.

The Perceptron algorithm works by calculating the dot product of x_i and W and adding the bias, b. The sign function is then passed the result and this is our activation function corresponding to the possible values of the outcome variable mentioned above. The sign function is defined as

$$sign(x) = \begin{cases} -1 & \text{if x} < 0 \\ 0 & \text{if x = 0} \\ +1 & \text{if x} > 0 \end{cases} \quad (1)$$

To determine if the perceptron has correctly classified the observation, the result of the sign function corresponds to the outcome variable possibilities of whether a patient has diabetes or not. Therefore, the whole process of making a prediction on an observation is defined as.

$$f(x) = sign((x_i \cdot W) + b) \quad (2)$$

Using the predictions from equation (2) the weight vector, W can be updated during the training process. By multiplying the prediction by the actual class of the observation and checking the resulting value is > 0 the update logic can

be simplified. If the result is > 0 then the perceptron has made a correct prediction and does not need to be updated. This can be seen in the logic table below showing if a correct prediction is made the result will always be > 0 and if an incorrect prediction is made the result will be $<= 0$

Truth	Prediction	Output
1	1	1
1	-1	-1
-1	1	-1
-1	-1	1

Table 1. Prediction vs truth logic table for predictions

If the perceptron makes an incorrect prediction during the training phase W is updated by multiplying y_i with x_i. Let W at the current time be defined as W_t. This update step is defined as

$$W_{t+1} = W_t + (y_i * x_i) \tag{3}$$

Similarly, b needs to be updated in the case of an incorrect prediction. Let b at the current time be defined as b_t. The update step is defined as

$$b_{t+1} = b_t + y_i \tag{4}$$

References

[1] Afshin Gholamy, Vladik Kreinovich, and Olga Kosheleva. Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation.

[2] Kaggle. Pima indians diabetes database — kaggle.com. `https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database`. [Accessed 10-09-2024].