Assignment on Perceptron

MTech in Applied AI
Deep Learning Techniques

by

Sanjeev Kumar Pandey[*]
Faculty: Dr Anamika Gupta

Submitted April 2, 2025

[*] Student ID: 31050, Enrolment No:MT23AAI001, Email: `pandey.sanjeev@yahoo.com`

Table of Contents

1 Perceptron

Write a simple python program to understand the working of a simple perceptron by implementing it for the logical OR gate, and analyze the effects of changing weights and bias on its output.

1.1 Python code

```
import numpy as np
import matplotlib.pyplot as plt
import itertools

# Define OR function dataset
X = np.array([[0, 0], [0, 1], [1, 0], [1, 1]])  # Inputs
y = np.array([0, 1, 1, 1])  # Expected output

# Define 10 different weight and bias combinations
weight_bias_combinations = [
    (-1, 1, -0.5),
    (1.1, 1.1, -0.6),
    (0.5, 0.5, -0.3),
    (1.5, -1.5, -0.7),
    (0.8, 0.8, -0.4),
    (1, -0.8, -0.5),
    (0.8, -1, -0.4),
    (1.2, 1.2, -0.6),
    (1.3, 1, -0.7),
    (-1, 1.3, -0.5)
]

# Function to plot decision boundaries
def plot_decision_boundaries():
    fig, axes = plt.subplots(2, 5, figsize=(15, 6))
    axes = axes.ravel()  # Flatten the 2D array of axes

    for idx, (w1, w2, b) in enumerate(weight_bias_combinations):
        ax = axes[idx]

        # Plot data points
        for i in range(len(X)):
            if y[i] == 1:
                ax.scatter(X[i][0], X[i][1], color='blue', marker='o', s=100, label="1" if i == 1 else "")
            else:
                ax.scatter(X[i][0], X[i][1], color='red', marker='x', s=100, label="0" if i == 0 else "")

        # Plot decision boundary: w1*x + w2*y + b = 0
        x_vals = np.linspace(-0.5, 1.5, 100)
        y_vals = (-w1 * x_vals - b) / w2  # Solve for y

        ax.plot(x_vals, y_vals, 'g-', label=f"w1={w1}, w2={w2}, b={b}")

        # Formatting
        ax.set_xlim(-0.5, 1.5)
        ax.set_ylim(-0.5, 1.5)
        ax.axhline(0, color='black', linewidth=0.5)
        ax.axvline(0, color='black', linewidth=0.5)
        ax.set_xlabel("x1", fontsize=8)
        ax.set_ylabel("x2", fontsize=8)
        ax.set_title(f"Config {idx+1}", fontsize=10)

        # Reduce font size in legend
        ax.legend(fontsize=7, loc="best")

        ax.grid(True)

    plt.tight_layout()
    plt.savefig('line_plot.png')
    plt.close()

# Call function to plot decision boundaries for all configurations
plot_decision_boundaries()
```

Listing 1: Code

1.1.1 Output

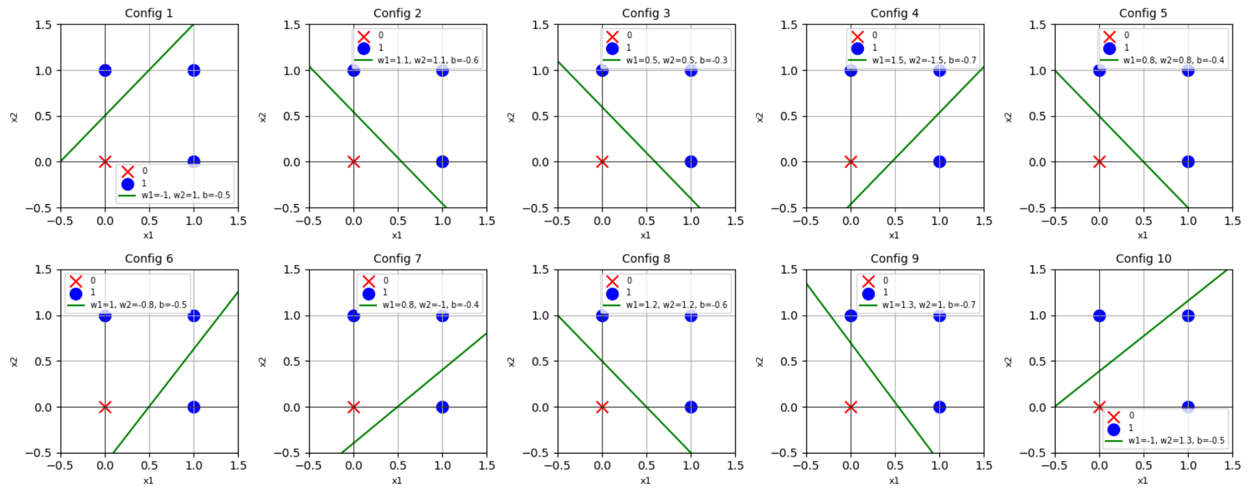

Figure 1: Separation Plan plotting using different weights and biases