Lesson 33: Perceptron

Data Science with Python – BSc Course

45 Minutes

Learning Objectives

The Problem: Traditional ML models have fixed architectures. How do neural networks learn? Understanding the perceptron is the first step toward deep learning.

After this lesson, you will be able to:

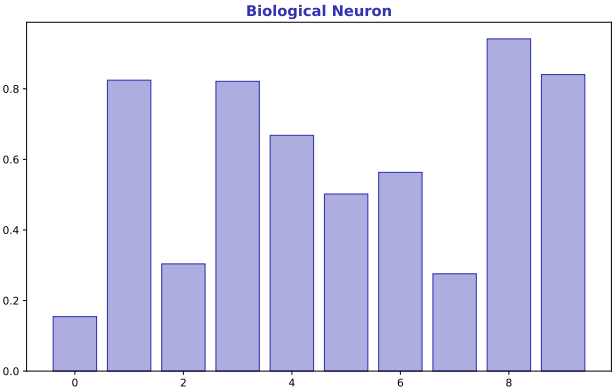
- Understand the biological inspiration for neural networks
- Build and train a single perceptron
- Recognize the limitation: only linearly separable problems
- Prepare for multi-layer networks that overcome this

Finance Application: Foundation for deep learning models in quantitative finance

Biological Neuron

Inspiration from the Brain

- Dendrites receive signals, soma processes, axon outputs
- Neuron "fires" when input exceeds threshold

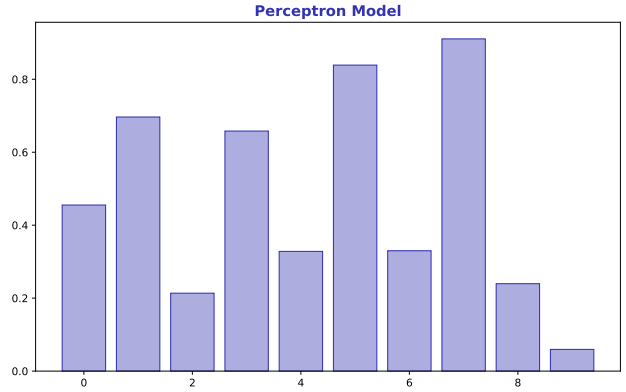


Perceptron: simplified mathematical model of biological neuron

Perceptron Model

The Simplest Neural Network

- Output: $y = \sigma(\sum w_i x_i + b)$ where σ is activation
- Weights w_i control importance of each input

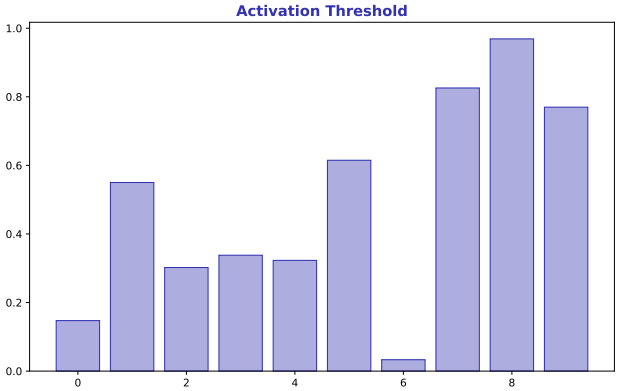


Single perceptron = logistic regression (same math, different framing)

Activation Threshold

When Does the Neuron Fire?

- Step function: output 1 if weighted sum > 0, else 0
- Modern: sigmoid, ReLU for smooth gradients

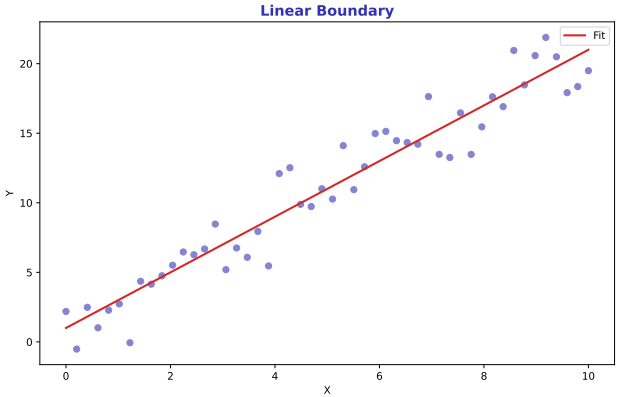


Step function: original perceptron. Sigmoid: smooth for gradient descent.

Linear Decision Boundary

What Can a Perceptron Learn?

- Single perceptron creates a linear boundary (hyperplane)
- Can separate AND, OR but not more complex patterns

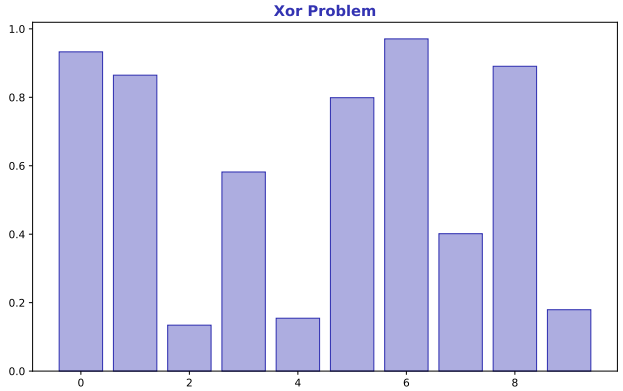


Perceptron = linear classifier. Same limitation as logistic regression.

The XOR Problem

A Famous Limitation

- XOR is not linearly separable – no single line can divide it
- This limitation stalled neural network research for years

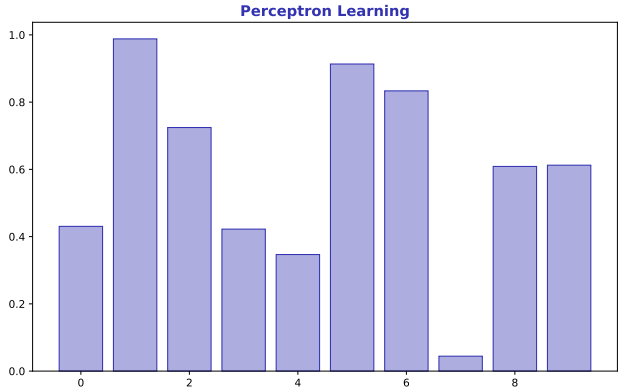


Solution: add hidden layers (multi-layer perceptron) to learn non-linear boundaries

Perceptron Learning Rule

How Weights Are Updated

- If wrong: $w_{new} = w_{old} + \eta \cdot (y_{true} - y_{pred}) \cdot x$
- Learning rate η controls step size

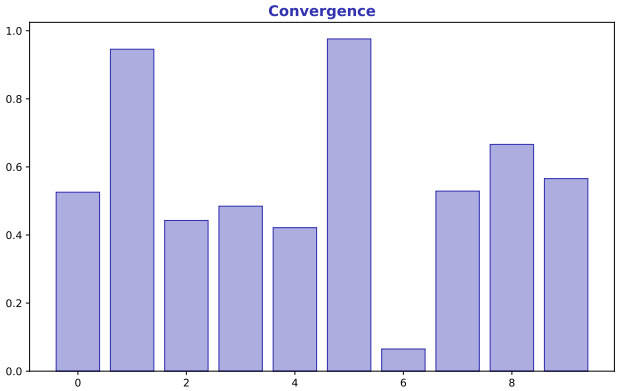


Perceptron learning rule: simple but powerful for linearly separable data

Convergence Guarantee

When Will Training Stop?

- If data is linearly separable, perceptron converges in finite steps
- If not separable, it oscillates forever (need multi-layer)

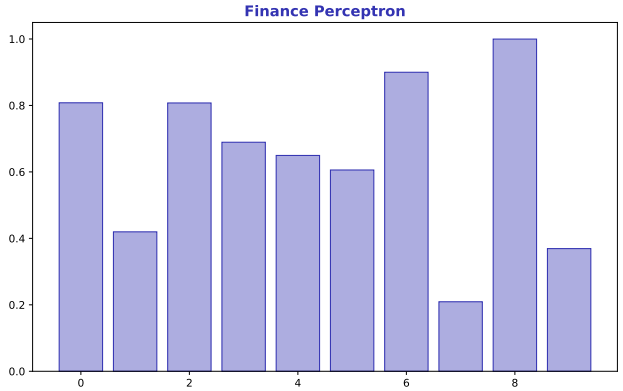


Perceptron Convergence Theorem (1962): guaranteed for separable problems

Finance Application

Simple Trading Signals

- Inputs: momentum, volatility, volume signals
- Output: buy (1) or sell (0) decision



Reality: Financial patterns are rarely linearly separable – need deeper networks

Hands-On Exercise (25 min)

Task: Implement Perceptron from Scratch

1. Create synthetic linearly separable 2D data
2. Implement perceptron learning rule in Python (no sklearn)
3. Train and plot decision boundary at each epoch
4. Try on XOR data – observe failure to converge
5. Compare to sklearn's `Perceptron` class

Deliverable: Animation of boundary evolution + XOR failure plot.

Extension: Modify learning rate – how does convergence speed change?

Lesson Summary

Problem Solved: We understand the building block of neural networks and its limitations.

Key Takeaways:

- Perceptron: weighted sum + activation = binary output
- Only learns linearly separable patterns
- XOR problem: needs hidden layers to solve
- Foundation for multi-layer perceptrons (MLPs)

Next Lesson: MLPs and Activations (L34) – adding hidden layers

Memory: Perceptron = single neuron = linear boundary. XOR needs hidden layers.