Lesson 35: Backpropagation

Data Science with Python – BSc Course

45 Minutes

Learning Objectives

The Problem: Neural networks have millions of weights. How does the network figure out which weights to adjust and by how much?

After this lesson, you will be able to:

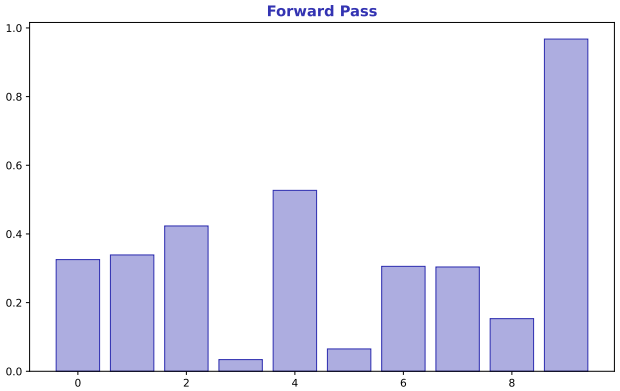
- Understand gradient descent optimization
- Interpret loss curves and diagnose training issues
- Configure learning rate and its effects
- Monitor training progress with validation metrics

Finance Application: Training predictive models on financial data

Forward Pass

Computing Predictions

- Input flows through network layer by layer
- Each layer: $z = Wx + b$, then $a = \sigma(z)$

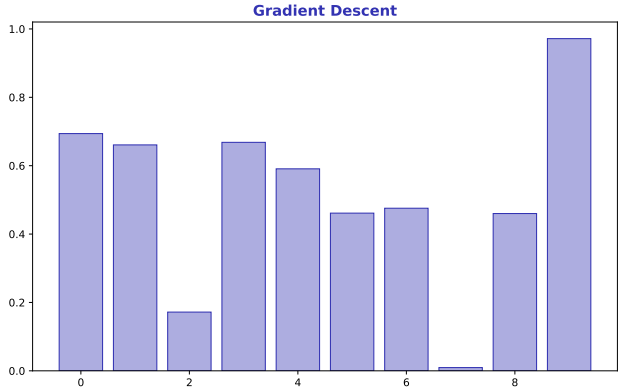


Forward pass: input → hidden(s) → output = prediction

Gradient Descent

Walking Downhill on the Loss Surface

- Compute gradient: $\frac{\partial L}{\partial w}$ tells direction of steepest increase
- Update: $w_{new} = w_{old} - \eta \cdot \frac{\partial L}{\partial w}$

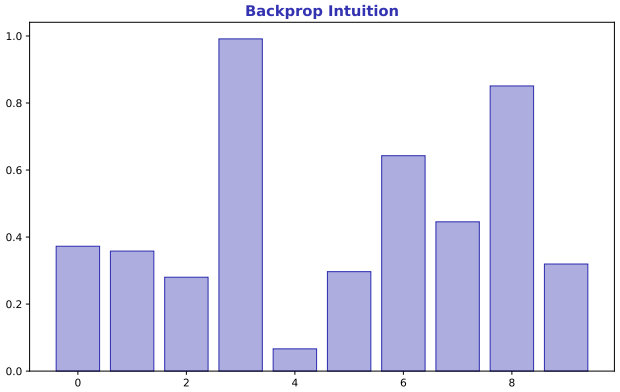


Gradient descent: repeatedly move opposite to gradient until minimum

Backpropagation Intuition

Efficiently Computing Gradients

- Chain rule: propagate error backward through layers
- Each layer's gradient depends on layers after it

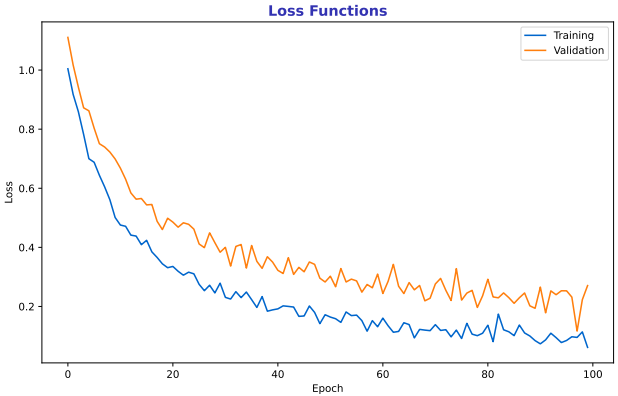


Backprop = efficient gradient computation via chain rule (not a learning rule)

Loss Functions

What Are We Minimizing?

- MSE: regression problems
- Cross-entropy: classification (binary or categorical)

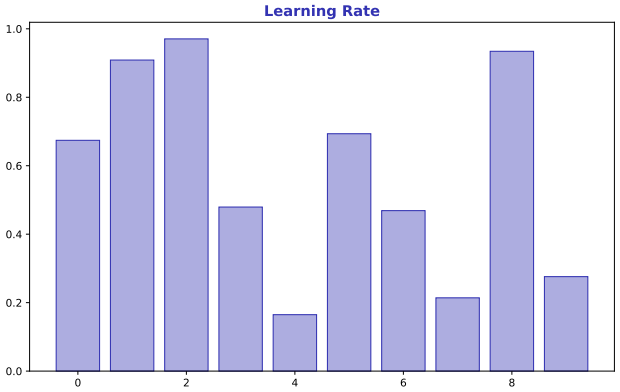


Match loss to problem: MSE for regression, cross-entropy for classification

Learning Rate

The Most Important Hyperparameter

- Too high: oscillates, may diverge
- Too low: converges very slowly

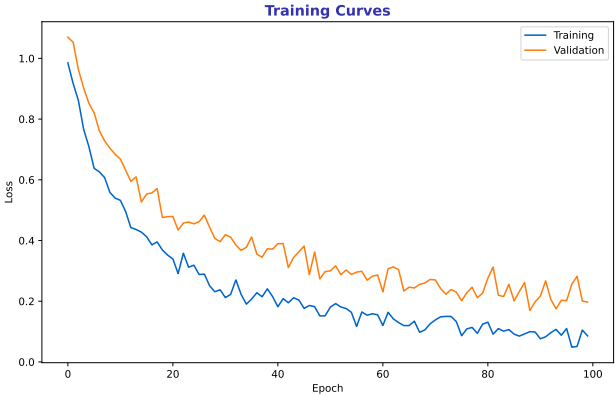


Start with 0.001 (Adam default). Reduce if loss is unstable.

Training Curves

Diagnosing Training

- Plot loss vs epochs for train and validation
- Train ↓, val ↓: good. Train ↓, val ↑: overfitting

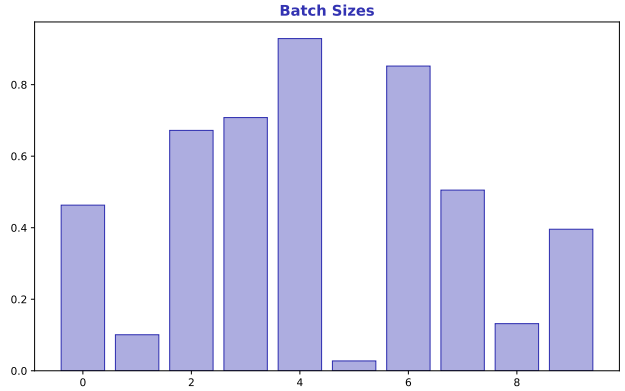


Always monitor validation loss – it shows real generalization

Batch Sizes

How Much Data Per Update?

- Batch: all data (slow, stable)
- Mini-batch: 32-256 samples (fast, noisy but works)
- Stochastic: 1 sample (very noisy)

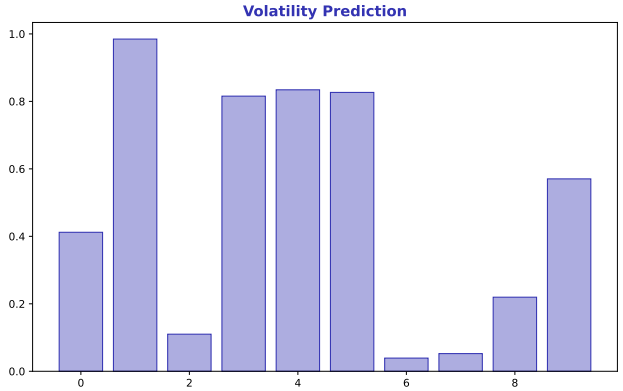


Default: mini-batch of 32. Larger batches need larger learning rates.

Volatility Prediction

Finance Application

- Input: lagged returns, volume, past volatility
- Output: next-day volatility (regression)



Monitor training curves carefully – financial data is noisy

Hands-On Exercise (25 min)

Task: Visualize Gradient Descent

1. Create simple 2D function (e.g., parabola) and plot surface
2. Implement gradient descent from scratch
3. Plot optimization path for different learning rates
4. Train Keras model and plot train/val loss curves
5. Try batch sizes 16, 64, 256 – compare convergence

Deliverable: 3D surface with optimization paths + loss curve comparison.

Extension: Implement momentum and compare to vanilla gradient descent

Lesson Summary

Problem Solved: We understand how neural networks learn through backpropagation and gradient descent.

Key Takeaways:

- Forward pass computes prediction, backward pass computes gradients
- Gradient descent updates weights to minimize loss
- Learning rate controls step size – crucial hyperparameter
- Monitor train/val loss curves to diagnose training

Next Lesson: Overfitting Prevention (L36) – regularization for neural networks

Memory: Backprop = chain rule. Learning rate = step size. Watch val loss.