Introduction to Neural Networks

From Brain to Business: How Machines Learn to Predict

Neural Networks for Business Applications

November 23, 2025

By the End of This Lecture, You Will Be Able To:

- **Explain** how biological neurons inspire artificial neural networks
- **Calculate** the output of an artificial neuron given inputs and weights
- **Design** a simple multilayer network architecture for a business problem
- **Trace** information flow through forward propagation
- **Describe** how networks learn by minimizing prediction errors
- **Evaluate** when neural networks are (and are not) appropriate for business predictions
- **Assess** the ethical implications of automated prediction systems

The Prediction Challenge: Can We Predict Markets?

The Business Question:

- Can we predict if a stock price will rise or fall tomorrow?
- Traditional methods: Statistical analysis, expert intuition, rule-based systems
- Challenge: Markets are complex, non-linear systems
- Many interacting factors: price history, volume, sentiment, volatility, interest rates

Why This Matters:

- Better investment decisions ($\rightarrow$ higher returns)
- Risk management ($\rightarrow$ protect capital)
- Portfolio optimization ($\rightarrow$ balanced exposure)
- Automated trading strategies ($\rightarrow$ scalability)

What We Need

A system that can:

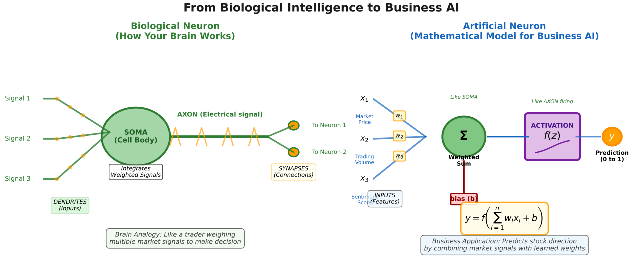
1. Process multiple inputs simultaneously
2. Learn patterns from historical data
3. Handle **non-linear** relationships
4. Improve predictions over time
5. Generalize to new market conditions

Inspiration from Nature

The human brain solves complex pattern recognition tasks every day by learning from experience. **Can we mimic this for business predictions?**

▶ Step 1: Understand how brains work

Nature's Computer: How Your Brain Makes Predictions



The Biological Neuron:

- **Dendrites:** Receive signals from other neurons
- **Soma (cell body):** Integrates incoming signals
- **Axon:** Transmits output to next neurons
- **Synapses:** Connections with varying strengths

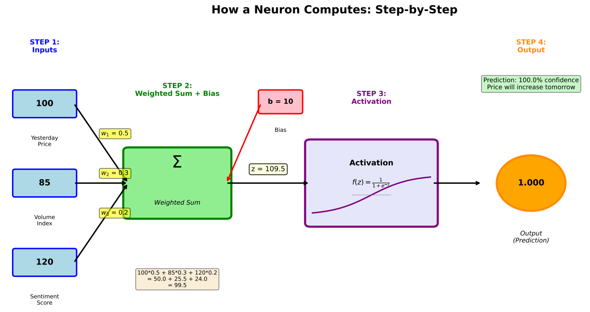
Key Insights for Business AI:

1. Multiple inputs combined → Consider many market factors
2. Weighted connections → Some factors matter more
3. Non-linear activation → Threshold effects (tipping points)
4. Layered processing → Abstract reasoning emerges

The Bridge to Mathematics

Just as your brain learned to predict patterns (faces, voices, market trends) through experience, we can create mathematical models that learn the same way!

The Artificial Neuron: From Biology to Mathematics



The Mathematical Model:

Step 1: Weighted Sum (like Soma)

$$z = \sum_{i=1}^{n} w_i x_i + b$$

- x_i: Input features (market data)
- w_i: Weights (learned from data)
- b: Bias (baseline adjustment)

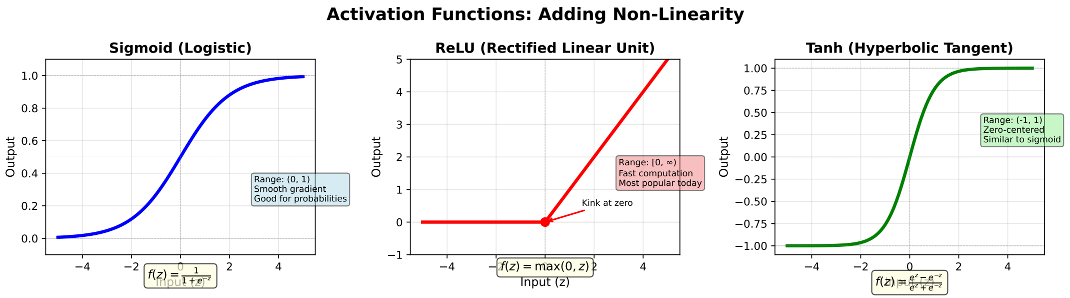
This mimics how dendrites weight incoming signals!

Step 2: Activation (like Axon)

$$y = f(z) = \frac{1}{1 + e^{-z}}$$

- f: Activation function (adds non-linearity)
- Output: probability between 0 and 1

Activation Functions: Why Non-Linearity Matters for Business



Why We Need Non-Linearity: Real business relationships are rarely straight lines!

- Diminishing returns: Doubling marketing spend doesn't double sales
- Threshold effects: Market sentiment must reach a tipping point to trigger buying
- Saturation: Customer engagement plateaus after certain point

Sigmoid

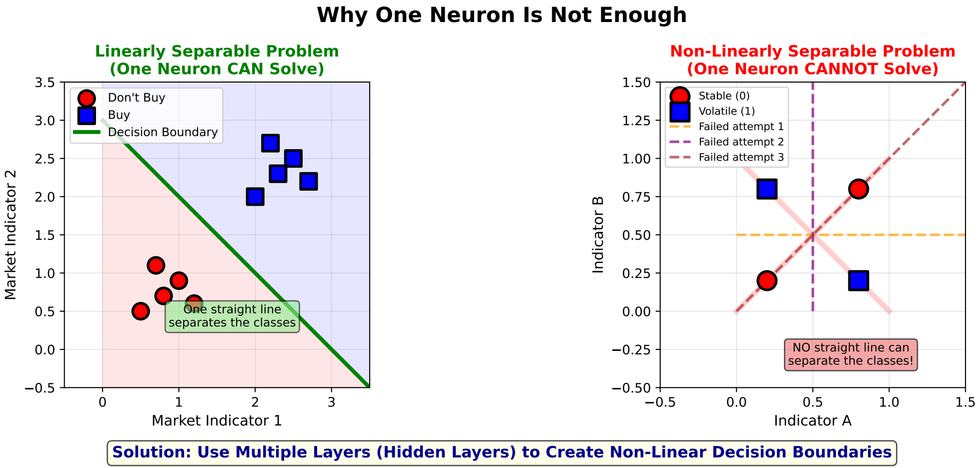
- Range: (0, 1) - perfect for probabilities
- Smooth gradients for learning
- Used in output for binary decisions

ReLU

- Fast computation, no saturation
- Most popular in modern networks
- Used in hidden layers for efficiency

Tanh

- Range: (-1, 1) - zero-centered
- Stronger gradients than sigmoid
- Alternative for hidden layers



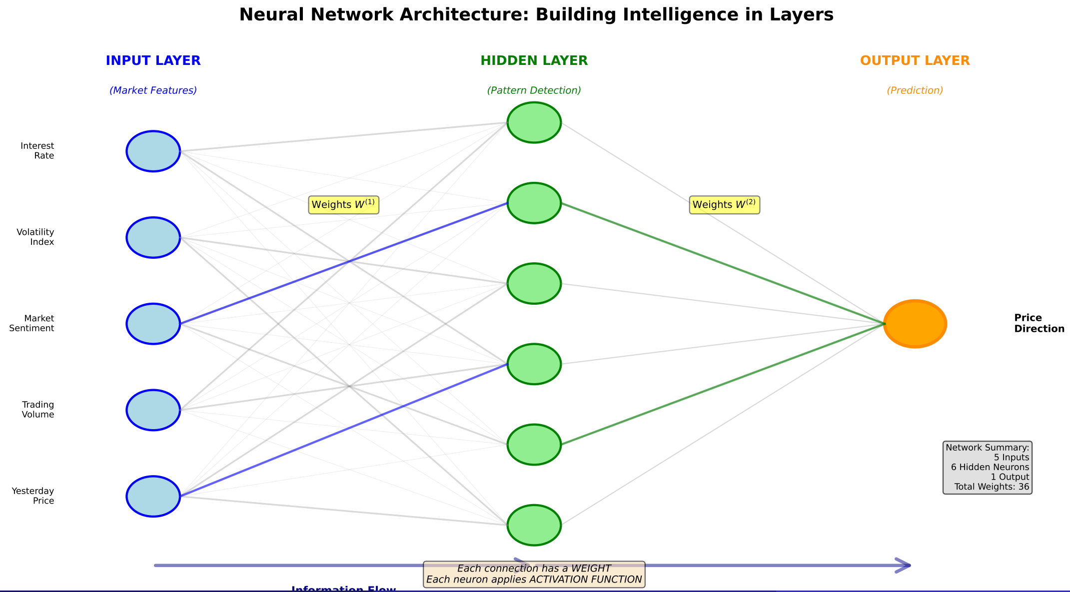
What One Neuron Can Do

- Draw a single straight line (hyperplane)

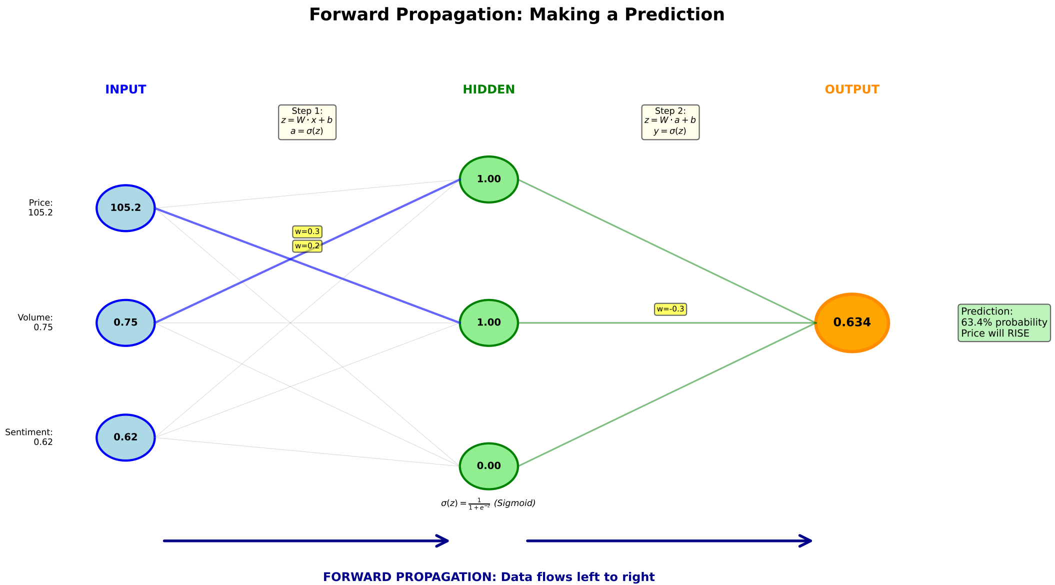
What One Neuron Cannot Do

- Complex, curved decision boundaries

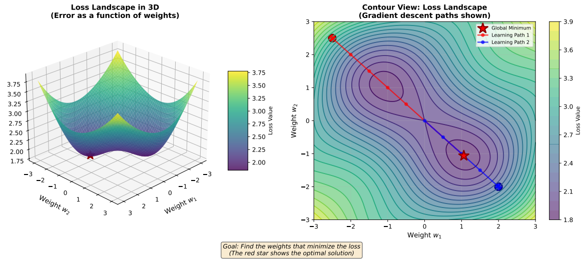
Building the Network: Layers of Intelligence

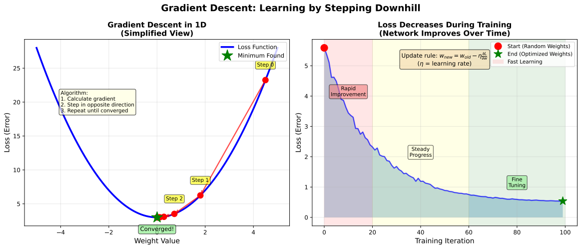


Forward Propagation: How Networks Make Predictions



Learning from Mistakes: How Networks Improve (Like Humans!)





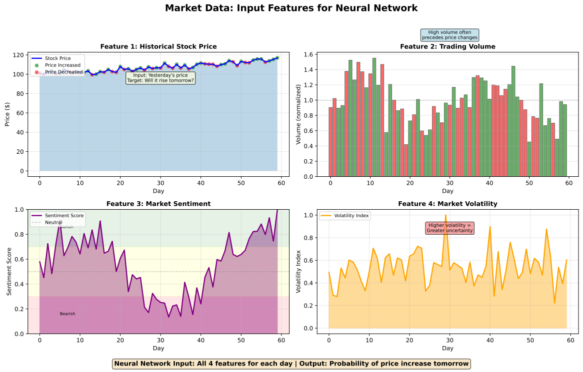
The Learning Process (Biological Parallel)

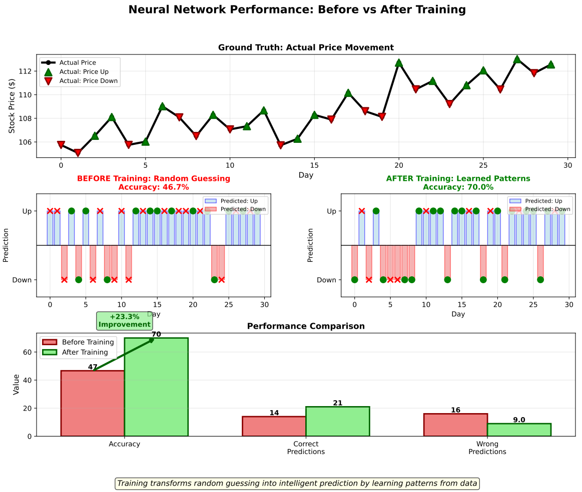
1. **Make prediction** → (like guessing in your brain)
2. **Measure error (Loss Function):** How wrong were we?

$$L = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

3. **Adjust weights (Gradient Descent):** Move toward better predictions

$$\partial L$$





Key Results & Takeaways

Summary: From Neurons to Predictions - And Beyond

Our Journey (Review):

1. Biology: Neurons integrate weighted signals
2. Mathematics: $y = f(\sum w_i x_i + b)$
3. Non-linearity: Activation functions enable complexity
4. Architecture: Layers learn hierarchical patterns
5. Forward prop: Making predictions
6. Learning: Gradient descent minimizes error
7. Application: Market prediction case study

When to Use Neural Networks:

- YES: Large datasets, complex patterns, non-linear relationships
- NO: Small data, need interpretability, simple linear problems

Important Limitations

- **Data hungry:** Need thousands of examples
- **Black box:** Hard to explain decisions to regulators
- **Overfitting:** May memorize training data, fail on new data
- **No guarantees:** Markets are inherently unpredictable
- **Computational cost:** Training requires significant resources

Ethical Responsibilities

- **Fairness:** Biased data $\rightarrow$ biased predictions $\rightarrow$ discriminatory outcomes
- **Transparency:** Can you explain decisions to stakeholders?
- **Accountability:** Who is responsible when AI makes wrong predictions?
- **Societal impact:** Automated trading can destabilize

How Gradient Descent Works: The Chain Rule
For a simple 2-layer network, we compute gradients layer-by-layer working backwards:

$$\frac{\partial L}{\partial w^{(2)}} = \frac{\partial L}{\partial y} \cdot \frac{\partial y}{\partial z^{(2)}} \cdot \frac{\partial z^{(2)}}{\partial w^{(2)}}$$

$$= (\hat{y} - y) \cdot \sigma'(z^{(2)}) \cdot a^{(1)} \quad \text{(output layer gradient)}$$

$$\frac{\partial L}{\partial w^{(1)}} = \frac{\partial L}{\partial y} \cdot \frac{\partial y}{\partial z^{(2)}} \cdot \frac{\partial z^{(2)}}{\partial a^{(1)}} \cdot \frac{\partial a^{(1)}}{\partial z^{(1)}} \cdot \frac{\partial z^{(1)}}{\partial w^{(1)}}$$

$$= (\hat{y} - y) \cdot \sigma'(z^{(2)}) \cdot w^{(2)} \cdot \sigma'(z^{(1)}) \cdot x \quad \text{(hidden layer gradient)}$$

Common Loss Functions:

- **Mean Squared Error (Regression):** $L = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ (measures average squared error)
- **Binary Cross-Entropy (Classification):** $L = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$ (measures probability mismatch)

Key Insight: Backpropagation efficiently computes how each weight contributed to the final error!

Appendix: Practical Considerations & Further Learning

Practical Tips for Building Business Neural Networks:

- **Start simple:** Try linear/logistic regression first as baseline
- **Feature engineering matters:** Good inputs > complex architecture
- **Avoid overfitting:** Use validation sets, regularization, dropout
- **Hyperparameter tuning:** Learning rate, architecture, batch size
- **Interpretability tools:** SHAP values, attention weights, feature importance

Recommended Resources:
Books:

- "Deep Learning" by Goodfellow, Bengio, Courville (comprehensive, mathematical)
- "Neural Networks and Deep Learning" by Michael Nielsen (free online, intuitive)
- "Hands-On Machine Learning" by Aurelien Geron (practical, code-focused)

Online Courses:

- Andrew Ng's Machine Learning (Coursera) - best introduction
- Fast.ai Deep Learning for Coders - practical, top-down approach
- MIT 6.S191 Introduction to Deep Learning - cutting-edge research

Tools: PyTorch, TensorFlow/Keras, scikit-learn
Practice Datasets: Kaggle competitions, Yahoo Finance API, UCI ML Repository

Appendix: Practice Problem for Business Students

Design Challenge: You are hired as a data scientist at a retail company.
Problem: Predict customer churn (will customer leave next month?)
Available Data:

- Customer demographics (age, location, income)
- Purchase history (frequency, recency, monetary value)
- Customer service interactions (calls, complaints, resolutions)
- Website engagement (visits, time spent, pages viewed)

Your Tasks:

1. Design a neural network architecture (how many layers? neurons?)
2. What would be your input features? (raw data or engineered features?)
3. What activation functions would you use and where?
4. What loss function is appropriate for this problem?
5. How would you evaluate model performance? (accuracy, precision, recall?)
6. What are potential ethical concerns with automated churn prediction?
7. How would you explain predictions to business stakeholders?

Discussion: Work in groups and present your design!