Exercise 10: Model Improvement Strategy

Estimated Time: 20-25 minutes

This exercise trains you to diagnose different model failure modes and recommend the right improvements without writing training code.

Here's the outline:

Together, the four tasks develop the kind of strategic thinking that's critical for building reliable, production-ready models.

Feel free to skip the scenario and goal if you prefer to jump straight to the tasks.

Scenario

You are the new ML lead at a fast-growing startup building a wine-quality prediction system. To speed up development, three sub-teams trained three different experimental variants of the model in parallel. These were meant to compete as candidates for deployment.

However, when you review the experiment dashboard this morning, all three models show different kinds of failures:

- **Model Variant A:** Shows excellent training performance, but validation performance deteriorates as training continues. The loss curves drift apart over time.
- **Model Variant B:** Training and validation metrics barely improve. Loss remains nearly flat across epochs.
- **Model Variant C:** Training repeatedly fails to complete. Logs show unstable behavior, with loss values jumping sharply between steps.

▸ **Dataset used for all variants: mstz/wine.**

Your job: Diagnose each model's failure mode and recommend specific improvements, choosing from techniques such as dropout, weight decay, learning-rate tuning, capacity adjustments, gradient clipping, or other relevant interventions.

You will not write code; instead, you will guide the team toward the right decisions and build a generalizable framework they can reuse.

Goal: Develop the ability to diagnose model failures, map problems to appropriate solution techniques, and build generalizable decision-making frameworks for improving ML models across architectures and domains.

Tasks

Part 1: Match Symptoms to Fixes

Complete the decision matrix below by checking (✓) which techniques help solve each problem. Some techniques may help multiple problems, while others are specific to one issue.

Problem Type	Dropout	Increase LR	Reduce LR	Add Layers	Gradient Clipping
Overfitting					
Underfitting					
Training Instability					
Slow Convergence					

> 💡 ***Hint:*** *Think about what each problem means. Overfitting = model is too flexible and memorizing. Underfitting = model is too simple. Training instability = updates are too aggressive. Slow convergence = learning steps are too conservative.*

Part 2: Analyze Model Variants

For each model variant below, review the provided metrics and loss curve, then **answer the questions**.

Note: Use your loss curve diagnosis skills from *Exercise 8*. The curve patterns will guide which category of techniques to recommend.

Model Variant A: Drifting Loss

Model: 3-layer MLP with architecture [128 → 64 → 32 → 1]

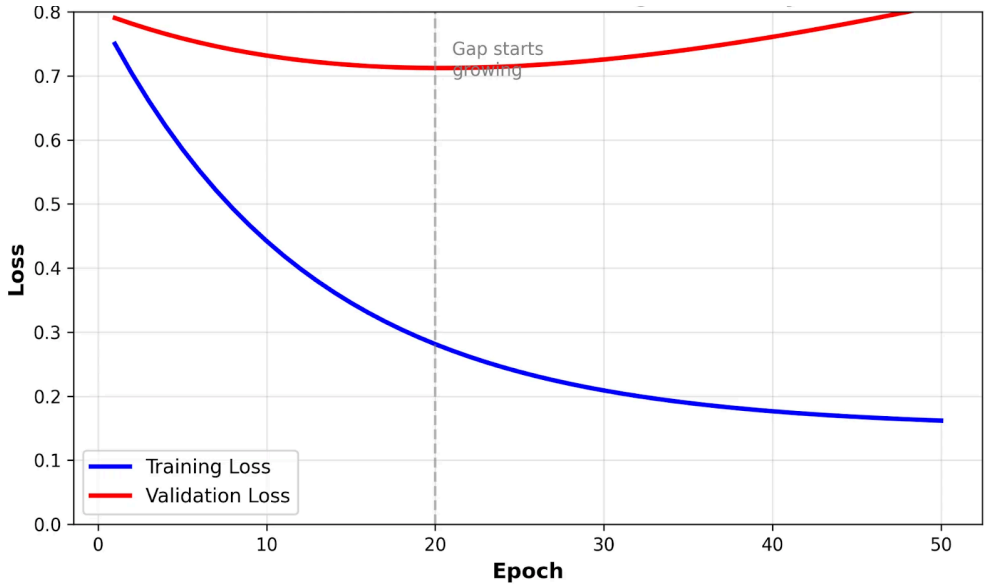
Dataset: 12,000 training samples, 3,000 validation samples

Loss function: BCEWithLogitsLoss (binary cross-entropy)

Optimizer: Adam, LR=1e-3, batch_size=64

Training: 100 epochs, NO regularization

Performance: Train accuracy: 92% / Validation accuracy: 73%, Train loss: 0.18 / Validation loss: 0.52



Questions:

1. What is the primary issue with this model? *(1 sentence)*

*_Write your answer here_*

2. Recommend 2-3 techniques in priority order with brief justification. *(2-3 sentences)*

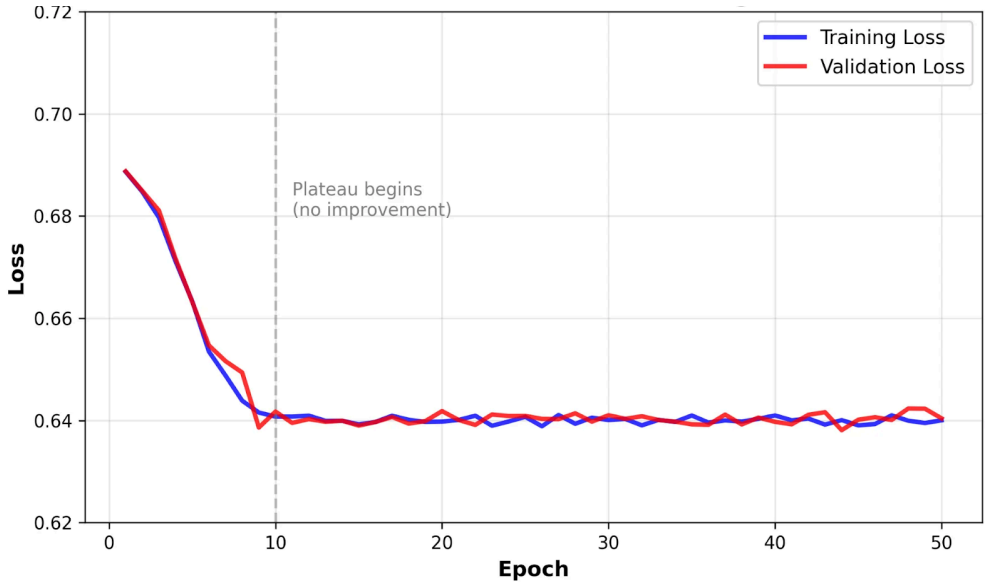
*_Write your answer here_*

3. Implementation: Show a 2-3 line code snippet demonstrating your top recommended technique.

*_Write your answer here_*

Model Variant B: Flat Loss

> **Model**: 2-layer MLP with architecture [64 → 32 → 1]
>
> **Dataset**: 8,000 training samples, 2,000 validation samples
>
> **Loss function**: BCEWithLogitsLoss (binary cross-entropy)
>
> **Optimizer**: Adam, LR=1e-3, batch_size=128
>
> **Training**: 100 epochs, no regularization applied
>
> **Performance**: Train accuracy 64% / Val accuracy 63%, Train loss 0.68 / Val loss 0.69



Questions:

1. What is the primary issue with this model? *(1 sentence)*

*_Write your answer here_*

2. Recommend 2-3 techniques in priority order with brief justification. *(2-3 sentences)*

*_Write your answer here_*

3. Implementation: Show a 2-3 line code snippet demonstrating your top recommended technique.

*_Write your answer here_*

Model Variant C: Unstable loss

Model: 4-layer MLP with architecture [256 → 128 → 64 → 32 → 1]

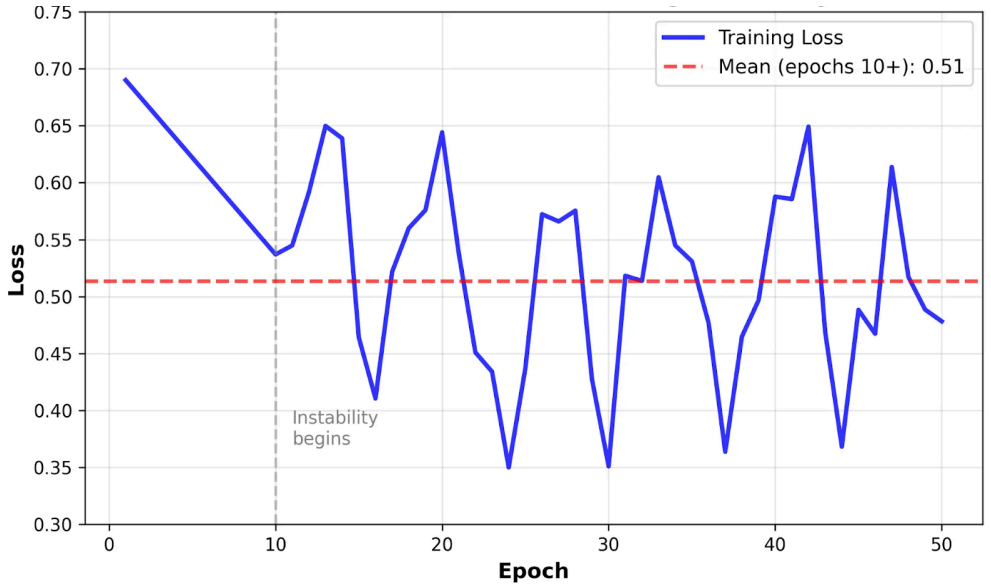
Dataset: 15,000 training samples, 5,000 validation samples

Loss function: BCEWithLogitsLoss (binary cross-entropy)

Optimizer: SGD, LR=0.1, batch_size=32

Training: 100 epochs attempted, training never stabilized

Performance: Loss oscillates between 0.3-0.7, never converging



Questions:

1. What is the primary issue with this model? *(1 sentence)*

*_Write your answer here_*

2. Recommend 2-3 techniques in priority order with brief justification. *(2-3 sentences)*

*_Write your answer here_*

3. Implementation: Show a 2-3 line code snippet demonstrating your top recommended technique.

*_Write your answer here_*

Part 3: Rank Techniques by Effort vs. Impact

Complete the table below by evaluating each technique's implementation effort and expected impact.

Technique	Implementation Effort	Expected Impact	Best Used When
Early Stopping	Low	Medium	Always (default practice)
Dropout			
Weight Decay			
LR Tuning			

Technique	Implementation Effort	Expected Impact	Best Used When
Collect More Data			
Architecture Search			

> 💡 ***Hint:*** *Implementation effort = how much code/time needed. Expected impact = how much improvement you'd expect. Think: Low/Medium/High for each.*

Key insight:

- Prioritize high-impact, low-effort techniques first.
- Save expensive options for when optimization techniques are exhausted.

Part 4: Build Your Improvement Playbook

Create a 5-step checklist that any team member could follow when facing an underperforming model. **Fill in each step below.**

Step 1: First, always check: *_Write your answer here_*

> 💡 ***Hint:*** *What foundational issues should you rule out before trying advanced techniques? (data quality, normalization, etc.)*

Step 2: If overfitting (train-val gap), try: *_Write your answer here_*

> 💡 ***Hint:*** *What's the easiest, most effective regularization technique to try first?*

Step 3: If underfitting (both losses high), try: *_Write your answer here_*

> 💡 ***Hint:*** *The model needs more capacity or more time to learn. What's the first thing to try?*

Step 4: Before collecting more data, ensure: *_Write your answer here_*

> 💡 ***Hint:*** *Have you extracted maximum value from your existing data? What optimization techniques are left?*

Step 5: Last resort options (expensive/time-consuming): *_Write your answer here_*

> 💡 ***Hint:*** *What requires significant resources or expertise? (data collection, architecture search, etc.)*

Key Takeaways

Model improvement is a systematic process, not random experimentation. By the end of this exercise, you should understand:

- How to diagnose model problems from loss curves and metrics
- Which techniques address which problems (and why)
- How to prioritize improvements based on effort vs. impact
- A reusable framework for tackling underperforming models

Remember: Start with high-impact, low-effort techniques (optimization and data improvements) before moving to expensive solutions. Change one thing at a time, observe results, then iterate.

▸ **Next Step:** Closing the Loop. You've diagnosed all three model variants and recommended specific improvements. What should each team do next to move from recommendations to deployment? *Consider model retraining, evaluation, and deployment trade-offs for final model selection.*

No need to answer, just a challenge for you to reflect on!