Instructor Solutions Guide V2

Discovery Worksheet: Introduction to ML & AI

Conceptual Answers and Discussion Prompts

October 2025

Instructor Notes

Purpose: Solutions for the conceptual discovery worksheet (v2). Focus on pattern recognition and reasoning over numerical calculations.

Time Allocation:

- Chart 1 (Overfitting): 8 minutes
- Chart 2 (K-Means): 8 minutes
- Chart 3 (Boundaries): 8 minutes
- Chart 4 (Gradient): 8 minutes
- Chart 5 (GANs): 8 minutes
- Chart 6 (PCA): 8 minutes
- Reflection: 5 minutes

Total: 45-55 minutes

Approach: Students discover patterns through visual analysis, not precise calculations.

Chart 1: Model Complexity - SOLUTIONS

Expected Answers

Task 1: Rank model complexity

Answer: A (Simple) → B (Balanced) → C (Most complex)

Reasoning: Model A is flat line (1 parameter), Model B is smooth curve (3 parameters), Model C wiggles through every point (15+ parameters).

Task 2: Identify performance pattern

Answer: **Model C** performs worst on orange test points.

Why: Model C fits training points perfectly but wildly misses test points. The curve oscillates dramatically between training points, showing it memorized noise rather than learned the underlying pattern.

Task 3: Explain the trade-off

Expected discovery: "Fitting training data too closely means the model learns the random noise in the data, not the true relationship. When new data arrives with different noise, predictions fail."

Common student responses:

- "Model C is too complicated" → Good! Connect to "unnecessary complexity"
- "Model C memorizes instead of generalizes" → Excellent! This is overfitting
- "The curve tries too hard" → Nice intuition, formalize as "minimizing training error too aggressively"

Task 4: Optimal complexity

Answer: **Model B**

Why: Model B balances training fit with smoothness. It misses some training points but captures the overall trend, leading to better predictions on new data.

Key Insight: The best model is NOT the one with lowest training error. Optimal complexity balances fit with generalization.
Formal term: *Bias-variance tradeoff*

- Model A: High bias (too simple)
- Model B: Balanced
- Model C: High variance (too complex, overfits)

Discussion Prompts

1. "If you were buying a house, which model would you trust? Why?"
2. "In what scenarios might Model A actually be better than Model B?" (Limited data, high noise)
3. "How would you know if your model is overfitting without test data?" (Cross-validation preview)

Chart 2: Clustering Algorithm - SOLUTIONS

Expected Answers

Task 1: Describe center movement

Answer: Centers move toward the average position of points assigned to them.

Pattern: Each center shifts to the centroid (geometric center) of its cluster.

Task 2: Observe point-to-center distances

Answer: Distances **decrease** from Step 0 to Step 5.

Algorithm objective: Minimize total distance from points to their cluster centers. Each iteration reduces this total distance until convergence.

Task 3: Identify assignment rule

Answer: Each point is assigned to its **nearest center**.

Mechanism: Calculate distance to all centers, choose minimum. This is the Voronoi partition principle.

Task 4: Detect convergence

Answer: Stop when centers no longer move (or move very little) between iterations.

Convergence criteria:

- Centers stabilize (change $<$ threshold)
- Cluster assignments stop changing
- Total distance stops decreasing

Key Insight: The algorithm alternates between two steps:

1. **Assignment**: Each point joins nearest center
2. **Update**: Each center moves to average of its points

This guarantees convergence to a local minimum of total within-cluster distance.
Formal term: *K-Means Clustering Algorithm*

Discussion Prompts

1. "What happens if you start with different initial center positions?" (May converge to different solutions)
2. "How would you choose the number of clusters (K)?" (Elbow method preview)
3. "Can this algorithm find non-circular clusters?" (No - motivates density-based methods)

Chart 3: Classification Boundaries - SOLUTIONS

Expected Answers

Task 1: Attempt linear separation

Answer: Datasets A, B, and C allow near-perfect separation with a straight line.

Verification: Students can mentally draw or sketch separating lines for these three.

Task 2: Identify impossible cases

Answer: **Dataset D** (XOR pattern)

Pattern: Red points at opposite corners, blue points at other two corners. No straight line can separate them.

Proof sketch: Any line that puts top-left red on one side will also put bottom-right red on the same side, but they need opposite classifications.

Task 3: Propose alternative solutions

Expected approaches:

- "Use two lines" → Good! Piecewise linear solution
- "Use a curve" → Yes! Nonlinear boundary (circle or polynomial)
- "Add a third dimension" → Excellent! Feature transformation (kernel trick preview)
- "Use regions instead of lines" → Tree-based methods preview

Mathematical proof (optional challenge):
Consider any line $ax + by = c$. The four XOR corners are: (0,0), (1,0), (0,1), (1,1).
If red points (0,0) and (1,1) satisfy $ax + by < c$, then:

- Point (0,0): $0 < c$ implies $c > 0$
- Point (1,1): $a + b < c$

But blue points (1,0) and (0,1) must satisfy $ax + by > c$:

- Point (1,0): $a > c$ implies $a > 0$
- Point (0,1): $b > c$ implies $b > 0$

From blue: $a > c$ and $b > c$, so $a + b > 2c$
From red: $a + b < c$
Contradiction: Cannot have $a + b > 2c$ and $a + b < c$ simultaneously.
Therefore, no linear separator exists. □

Key Insight: Linear models work when classes are linearly separable. Some patterns require nonlinear decision boundaries.
Formal terms:

- *Linear separability* - can separate with hyperplane
- *XOR problem* - classic non-linearly separable case
- *Feature transformation* - project to higher dimension where linear separation works

This limitation motivated development of neural networks and kernel methods.

Discussion Prompts

1. "Real-world example of XOR-like pattern?" (Mutual interaction effects: medicine A helps, medicine B helps, but A+B together harmful)
2. "How do neural networks solve this?" (Hidden layer creates feature transformation)
3. "Trade-off of using curves vs lines?" (Flexibility vs complexity)

Chart 4: Optimization Paths - SOLUTIONS

Expected Answers

Task 1: Compare final destinations

Answer: **No**, Path A (red) and Path B (blue) end in different valleys.

Implication: Starting position matters! Different initializations lead to different solutions.

Task 2: Identify the better solution

Answer: **Path B** finds the lower valley.

Global minimum? **Yes**, Path B reaches the lowest point on the entire landscape.

Path A gets trapped in a *local minimum* - lowest in its region but not globally optimal.

Task 3: Analyze step size effects

Steps too large: "The algorithm might overshoot valleys, bouncing around without settling. Could skip over good solutions or fail to converge."

Steps too small: "Progress is very slow. Takes many iterations to descend. May get stuck near starting point or take impractically long to reach minimum."

Task 4: Propose improvement strategies

Expected strategies:

- "Try multiple random starting points" → Yes! Random restart strategy
- "Use larger initial steps, then smaller steps" → Learning rate scheduling
- "Add randomness to escape local minima" → Simulated annealing, stochastic methods
- "Look ahead before committing to step" → Momentum, adaptive methods

Key Insight: Gradient-based optimization follows steepest descent but can get trapped in local minima. Step size (learning rate) controls speed vs stability trade-off.
Formal terms:

- *Gradient descent* - follow negative gradient downhill
- *Learning rate* - step size parameter
- *Local vs global minimum* - best nearby vs best overall
- *Convex optimization* - landscapes with single global minimum (no local traps)

Discussion Prompts

1. "Why is initialization random in practice?" (Systematic bias avoided, explore solution space)
2. "How do neural networks handle millions of parameters with this approach?" (Stochastic gradient descent)
3. "Can we guarantee finding global minimum?" (Only for convex problems; otherwise use heuristics)

Chart 5: Adversarial Training - SOLUTIONS

Expected Answers

Task 1: Track quality progression

Answer: Epoch 1 shows random noise blob. Epoch 100 shows structured, realistic data with clear shape and low noise.

Pattern: Quality improves systematically from 0% realism to 95% realism over 100 epochs.

Task 2: Analyze competitive dynamics

Phase 1 (Epochs 1-10): **Discriminator** has the advantage (low loss, easily detects fakes).

Phase 3 (Epochs 50-100): Both loss curves **converge to equilibrium**, oscillating around similar values near the equilibrium point.

Task 3: Explain mutual improvement

Expected discovery: "Competition drives improvement for both. When Discriminator gets better at detecting fakes, Generator must improve to fool it. When Generator creates better fakes, Discriminator must sharpen detection skills. Each forces the other to improve."

Analogy: Arms race between counterfeiters and detectives - both get better over time through competition.

Task 4: Predict equilibrium outcome

Answer: The Generator has learned to create **perfect imitations** indistinguishable from real data.

At equilibrium: Discriminator can only guess randomly (50% accuracy). Generator produces realistic outputs that pass all tests.

Key Insight: Adversarial training uses competition between two neural networks:

- **Generator** creates synthetic data
- **Discriminator** detects fakes

Both improve until equilibrium (Nash equilibrium) where generated data becomes indistinguishable from real data.
Formal term: *Generative Adversarial Network (GAN)*
Applications: Image generation, data augmentation, style transfer, deepfakes

Discussion Prompts

1. "Ethical concerns with perfect fake generation?" (Deepfakes, misinformation, authentication challenges)
2. "What prevents Generator from memorizing training data?" (Discriminator penalizes copies; noise input forces generalization)
3. "Other domains where adversarial training applies?" (Security, game AI, reinforcement learning)

Chart 6: Dimensionality Reduction - SOLUTIONS

Expected Answers

Task 1: Assess data structure

Answer: Points lie **near a flat surface** (plane) within the 3D space, not filling entire volume.

Observation: Data has lower intrinsic dimensionality than ambient space. The third dimension is mostly redundant.

Task 2: Evaluate compression feasibility

Answer: **Yes**, compression is feasible without major information loss.

Why: Since points lie near a 2D plane embedded in 3D space, we can describe positions using only 2 coordinates on that plane instead of 3 coordinates in full 3D space.

Typical information retention: 98-99% of variance captured with 33% reduction in dimensions.

Task 3: Analyze reconstruction quality

Answer: Reconstructed points (red) are **very close** to original positions (blue).

Quality indicator: Small reconstruction errors mean the 2D representation preserves most information. The compression is nearly lossless for this dataset.

Task 4: Generalize the principle

Expected discovery: "High-dimensional data can be compressed when dimensions are correlated or when data lies near a lower-dimensional structure (manifold). If variations are mostly along certain directions, we can ignore perpendicular directions with minimal information loss."

Conditions for effective compression:

- Correlated features (redundancy)
- Data concentrated near lower-dimensional manifold
- Some dimensions have much higher variance than others

Key Insight: Dimensionality reduction projects high-dimensional data onto principal directions of maximum variance, discarding low-variance directions.
Formal term: *Principal Component Analysis (PCA)*
Benefits:

- Reduced storage (fewer numbers per point)
- Faster computation (lower-dimensional algorithms)
- Visualization (project to 2D/3D for plotting)
- Noise reduction (low-variance directions often noise)

Trade-off: Slight information loss vs significant dimensionality reduction

Discussion Prompts

1. "When would PCA fail to compress effectively?" (Uniformly distributed data, uncorrelated dimensions)
2. "How many dimensions to keep?" (Scree plot, cumulative variance threshold)
3. "Non-linear dimensionality reduction?" (t-SNE, UMAP for manifold learning)

Connections Across Charts - SOLUTIONS

Recurring Patterns

1. Optimization objectives

- Chart 1: Minimizing *prediction error (with regularization to prevent overfitting)*
- Chart 2: Minimizing *total within-cluster distance*
- Chart 4: Minimizing *error/loss function*

2. Complexity trade-offs

General principle: *"Simpler models generalize better, but must be complex enough to capture true patterns. Optimal complexity balances fit with parsimony."*

Examples:

- Chart 1: Too simple = high bias, too complex = high variance
- Chart 3: Linear simple but fails on XOR; nonlinear flexible but risks overfitting
- Chart 6: More dimensions = more detail, but redundancy increases noise

3. Iterative improvement

Charts with iterative algorithms: **2 (K-means), 4 (Gradient descent), 5 (GAN training)**

Common pattern: Start with initial guess, improve incrementally via local updates, converge when updates become small.

Key Insights (Expected)

Students should identify 3 major themes:

1. **Generalization over memorization**: Models must capture patterns, not noise. Training performance doesn't guarantee test performance.
2. **Iterative optimization**: Most ML algorithms improve solutions gradually through repeated updates rather than finding optimal solutions directly.
3. **Trade-offs are fundamental**: Complexity vs simplicity, accuracy vs interpretability, compression vs information preservation. No free lunch.

Questions for Lecture (Examples)

1. "How do we choose between different models when trade-offs exist?"
2. "Can we prove an algorithm will converge, or do we rely on empirical observation?"
3. "What mathematical frameworks unify these different concepts?"
4. "How do these principles scale to modern deep learning with millions of parameters?"
5. "When should we use simple models despite having powerful complex models available?"

Lecture Integration Strategy:

- Reference specific charts when introducing formal concepts
- Use student discoveries as motivation for mathematical formulations
- Validate student intuitions with rigorous derivations
- Extend patterns to broader ML theory (statistical learning, optimization theory)
- Connect to modern applications (deep learning, production systems)

Assessment: Students who completed worksheet thoroughly will have 70% conceptual foundation before lecture begins. Lecture formalizes and extends discoveries rather than building from zero.