ML Lab 3 Answers

1. Reducing to 100 Iterations

Observation:

- Initial cost (at theta=[0;0]) remains ~32.07
- Final theta values show minimal change from initial [0;0]
- Linear regression line stays nearly flat
- Predictions become less accurate:

For population=35,000: ~$0-5,000 profit

For population=70,000: ~$0-10,000 profit` ` `

Explanation:
Gradient descent hasn't had enough steps to:

- Properly descend the cost function surface
- Adjust theta parameters meaningfully
- Reach the convex bowl of the cost function

2. Increasing to 10,000 Iterations

Observation:

- Cost decreases further (beyond original 1500-iteration value)
- Theta converges closer to optimal values:

Theta ≈ [-3.63; 1.16] (vs original [-3.88; 1.19])` ` `

- Regression line fits data better
- Predictions become more stable:

For 35,000: ~$45,000 profit

For 70,000: ~$91,000 profit` ` `

Explanation:
Additional iterations allow:

- More precise convergence to minimum
- Better parameter optimization
- Smoother traversal down cost surface
- Stabilization of gradient updates

Key Visual Differences

1. **Cost Function Plot:**
    - 100 iterations: Contour plot shows theta far from center
    - 10k iterations: Theta marker reaches deepest contour level
2. **Learning Curve:**
    (Plot J_history vs iterations)
    - 100 iterations: Curve still descending sharply
    - 10k iterations: Curve flattens showing convergence

Technical Insight

The gradient descent update rule in gradientDescent.m:

theta(1) = theta(1) - (alpha/m) * temp0;

theta(2) = theta(2) - (alpha/m) * temp1;

Works better with:

- **More iterations:** Allows smaller, precise steps near minimum
- **Learning rate (α=0.01):** Prevents overshooting with high iterations

Pro Tip: For this dataset, 1500-3000 iterations with α=0.01 typically suffices. Extreme values (>1M iterations) show diminishing returns due to floating-point precision limits.