Machine Learning B (2025)
Home Assignment 3

Bar Segal xsb740

13/05/2025

Contents

1 SVM with Kernels (30 points)

Question 1: Linear SVM Training Loss

We trained a linear SVM with margin parameter $r = 1$ and misclassification penalty $C \in \{1, 100, 1000\}$, and computed the average hinge loss on the training set.

Table 1: Hinge loss for Linear SVM ($r = 1$).

C	Hinge Loss
1	0.800000
100	0.800000
1000	0.800001

Explanation: The loss plateaus around 0.8 for all tested penalties. Because the dataset is not linearly separable, most margin violations already occur at $C = 1$; increasing C cannot correct them and therefore yields only negligible improvement.

Question 2: Gaussian-Kernel SVM Training Loss

We trained an RBF-kernel SVM with fixed misclassification penalty $C = 1$ and kernel width parameter $a \in \{0.1, 1, 10\}$. $K(x, x') = \exp(-\|x - x'\|^2/(2a^2))$.

Table 2: Hinge loss for Gaussian-kernel SVM ($C = 1$).

a	Hinge Loss
0.1	0.062703
1.0	0.726238
10.0	0.799991

Explanation: A narrow kernel width ($a = 0.1$) produces the lowest loss (0.062), confirming that a highly non-linear boundary is needed. As a grows, the kernel becomes smoother and the model approaches the linear regime, so the loss rises toward the linear SVM baseline.

2 Logistic regression on MNIST (70 points)

Question 3: Training Loss over Iterations

I ran full–batch gradient descent (GD) and mini–batch SGD (batch size $b = 10$, $\gamma = 0.001$) for 100 iterations and recorded the training loss every 10 iterations (see Figure 1).

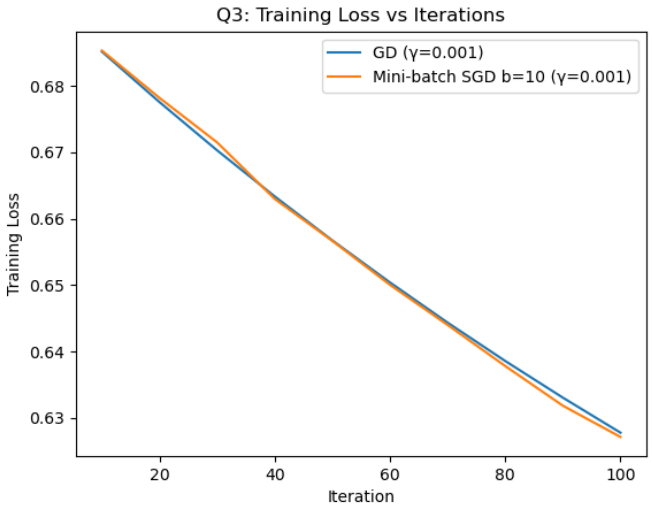


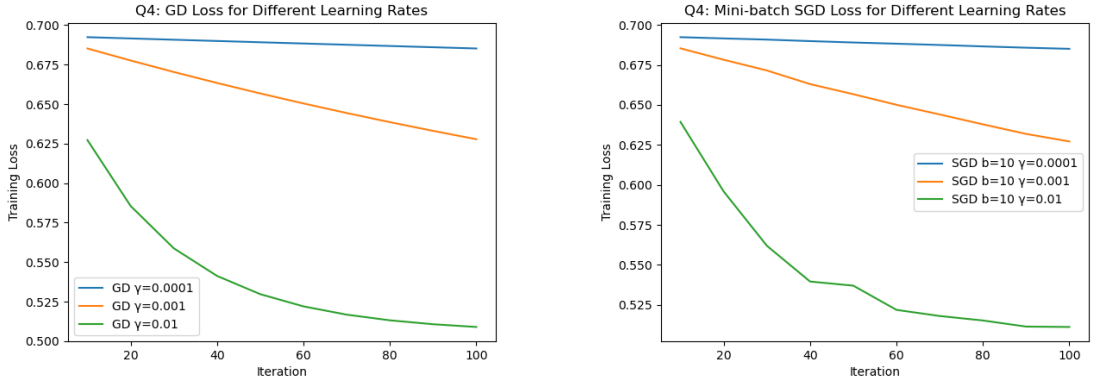
Figure 1: Training loss vs. iteration for GD and mini–batch SGD ($b = 10$, $\gamma = 0.001$).

Conclusion: Both GD and mini–batch SGD converge steadily at $\gamma = 0.001$, with GD being slightly smoother and both more or less matching the final loss by iteration 100.

Question 4: Effect of Learning Rate γ

Figure 2 shows training loss curves for GD and mini–batch SGD ($b = 10$) under three learning rates $\gamma \in \{0.0001, 0.001, 0.01\}$. As γ increases, convergence accelerates:

- $\gamma = 0.0001$: very slow decrease - $\gamma = 0.001$: moderate progress - $\gamma = 0.01$: fastest decline, reaching the lowest loss by iteration 100



(a) Full–batch GD

(b) Mini–batch SGD ($b = 10$)

Figure 2: Training loss vs. iteration for $\gamma = 0.0001, 0.001, 0.01$. The largest rate (0.01) converges most quickly.

About differences between GD and mini-batch SGD: The mini-batch curves show the same ordering as GD: $\gamma = 0.01$ converges the fastest and lowest. Stochasticity adds visible noise, especially for $\gamma = 0.01$, but the trend is monotone downward, whereas $\gamma = 0.0001$ barely moves.

Question 5: Impact of Batch Size on Convergence

Using the best constant rate $\gamma = 0.01$, Figure 3 compares mini–batch SGD convergence for $b \in \{1, 10, 100\}$. Larger batches yield smoother, more stable descent, especially smoother for $b = 100$; very small batches ($b = 1$) fluctuate substantially but still approach similar final loss, although not quite as good.

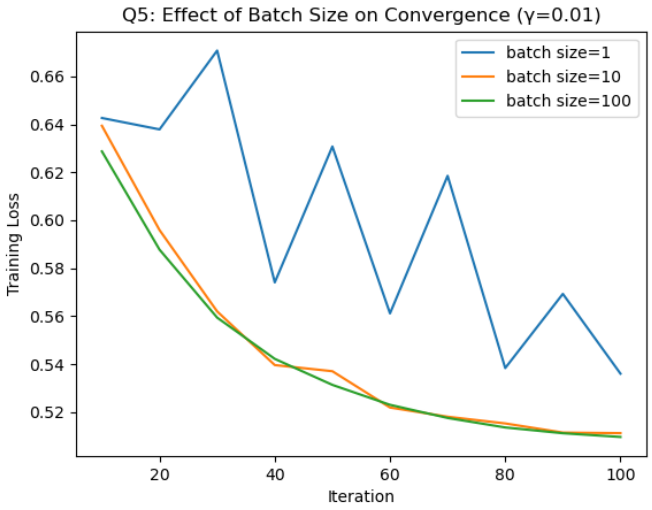


Figure 3: Training loss vs. iteration for batch sizes $b = 1$, 10, 100 at $\gamma = 0.01$.

Conclusion: Using very small batches (e.g., 1) makes each gradient step cheap but noisy, so the loss "zig-zags" and occasionally climbs before heading downward; this variance can slow overall convergence even though you take many quick steps. Increasing the batch to 10–100 gives a more reliable gradient estimate, so the curve is smoother and descends almost monotonically, letting you reach a lower loss in fewer effective updates (at the cost of a bit more work per update). In short, larger batches trade a higher per-iteration cost for steadier, more predictable progress.

Question 6: Constant vs. Diminishing Learning Rate

Figure 4 plots mini–batch SGD ($b = 10$) with constant $\gamma = 0.01$ versus a diminishing schedule $\gamma_t = 1/t$. The diminishing schedule achieves a quicker early loss reduction and marginally lower final loss, indicating improved stability as training proceeds.

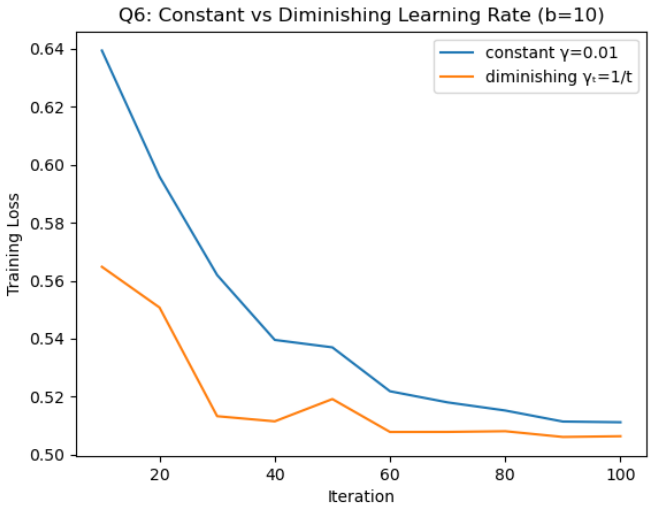


Figure 4: Training loss vs. iteration for constant $\gamma = 0.01$ (blue) and diminishing $\gamma_t = 1/t$ (orange), $b = 10$.

Conclusion: A diminishing learning rate $\gamma_t = 1/t$ outperforms the fixed $\gamma = 0.01$ by converging faster at first and yielding a slightly lower loss by iteration 100. Hence, the diminishing schedule $\gamma_t = 1/t$ not only converges faster early on and ends with a lower loss, but it also edges out the previously best constant rate $\gamma = 0.01$ from Question 4, delivering a modest yet measurable performance gain.