CS109 Lecture 20: LLM Learning Guide

Logistic Regression

Summer 2026

Lecture 20: LLM Learning Guide

Before lecture: read the Lecture 20 slides (logistic regression). Then open your favorite LLM and work through the concepts below **in order**. Each concept has a *Learn* prompt and a *Test me* prompt. Attempt the "Test me" questions yourself before asking the LLM to grade you.

How to get the most out of this:

- Tell the LLM your level: "I'm a student in Stanford's CS109 (probability for computer scientists). Explain at that level."
- After any answer, ask "why?" at least once and ask it to show each step.
- When it states a formula, ask it to define every symbol before continuing.
- Attempt practice yourself first, then say: "Here is my reasoning: _____. Where exactly am I wrong, if anywhere?"

Concept 1: The classification setup

Learn: > "Set up the supervised binary classification problem the way CS109 does. I have n i.i.d. training datapoints; each datapoint has m features $x_1, \ldots, x_m$ and a single binary output y. Explain that the goal is a machine that takes x in and returns $P(Y = 1 \mid X = x)$, and that we then threshold that probability to get a class prediction. Walk through three concrete examples (movie recommendation from which movies a user liked, heart-abnormality diagnosis from tomography regions, ancestry from SNPs) and, for each, say what one row of the training table is. Then explain the difference between classification (categorical output) and regression (real-valued output)."

Test me: > "Describe several prediction scenarios and make me say for each: is this classification or regression, what is a single datapoint, what are the features, what is y, and what would the model output. Push back if I confuse the prediction $\hat{y} \in [0, 1]$ with the class label."

Concept 2: The sigmoid function

Learn: > "Explain the sigmoid (logistic) function $\sigma(z) = \frac{1}{1+e^{-z}}$: its shape, that it squashes any real number into $(0, 1)$, that $\sigma(0) = 0.5$ is the inflection point, its behavior as $z \to \pm\infty$, and why that makes it a natural way to turn a weighted sum into a probability. Warn me that this σ has nothing to do with the standard deviation σ. Then derive the identity $\sigma'(z) = \sigma(z)[1 - \sigma(z)]$ step by step — I will need it later."

Test me: > "Quiz me on sigmoid values and reasoning: given z, what is the approximate $\sigma(z)$; given a target probability, what sign must z have; and make me derive $\sigma'(z) = \sigma(z)(1 - \sigma(z))$ from scratch without looking. Then ask me why thresholding $\sigma(z)$ at 0.5 is the same as thresholding z at 0."

Concept 3: The logistic regression assumption

Learn: > "State the logistic regression assumption precisely: $P(Y = 1 \mid X = x) = \sigma(\theta^T x) = \sigma\left(\sum_{j=0}^{m} \theta_j x_j\right)$, where we define $x_0 = 1$ so that θ_0 acts as an intercept. Explain what the weighted sum z means intuitively (evidence for $Y = 1$), what a large positive, large negative, and zero weight each say about a feature, and why the decision boundary $\theta^T x = 0$ is a line/hyperplane — making logistic regression a *linear* classifier. Also explain why the name is unfortunate: it is a classification algorithm, not a regression algorithm."

Test me: > "Give me a trained θ vector and a datapoint and make me compute z, then $\sigma(z)$, then the class prediction. Then make me write the equation of the decision boundary and sketch it. Finally, quiz me on interpreting individual weights when all features are binary."

Concept 4: The log-likelihood of the training data

Learn: > "Derive the logistic regression log-likelihood. Start from the Bernoulli PMF written in the differentiable form $P(Y = y) = \hat{y}^{\,y}(1 - \hat{y})^{1-y}$ where $\hat{y} = \sigma(\theta^T x)$; explain the trick of why that expression equals $\hat{y}$ when $y = 1$ and $1 - \hat{y}$ when $y = 0$. Then use the i.i.d. assumption to get the likelihood of the whole dataset as a product, take the log to get >

$$LL(\theta) = \sum_{i=1}^{n} y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}),$$

> and explain that this is *exactly* maximum likelihood estimation — the same recipe as the previous lecture, just with a richer model. Mention that this quantity is called cross-entropy loss in machine learning, and that LL is concave in θ."

Test me: > "Give me a tiny labelled dataset (3 or 4 points) and a candidate θ, and make me compute the log-likelihood of the data by hand. Then give me a second θ and ask which one the data prefers. Then ask me: where exactly did I use the i.i.d. assumption, and why do we take the log?"

Concept 5: The gradient, and where it comes from

Learn: > "Derive the logistic regression gradient $\frac{\partial LL}{\partial \theta_j} = \sum_{i=1}^{n} x_j^{(i)} \left[y^{(i)} - \sigma(\theta^T x^{(i)})\right]$ in full. Do it for a single training example first, using the chain rule in two pieces: (1) $\frac{\partial LL}{\partial \hat{y}} = \frac{y - \hat{y}}{\hat{y}(1-\hat{y})}$ and (2) $\frac{\partial \hat{y}}{\partial \theta_j} = \hat{y}(1 - \hat{y}) x_j$, and show me the beautiful cancellation when you multiply them. Then explain why the gradient for the whole dataset is just the sum of the per-example gradients. Finish by interpreting the formula: $y - \hat{y}$ is the error, and x_j decides how much of the blame feature j receives."

Test me: > "Make me reproduce the entire derivation from scratch with no hints, grading every algebra step like a CS109 grader. Then quiz me on the interpretation: what happens to the update when the model is already confidently correct, and what happens when $x_j = 0$ for a datapoint?"

Concept 6: Gradient ascent and the training loop

Learn: > "Explain gradient ascent as an optimization algorithm: initialize θ, repeatedly compute the gradient of LL, and step $\theta_j \mathrel{+}= \eta \cdot \frac{\partial LL}{\partial \theta_j}$. Explain the role of the step size η (too large: overshoot and oscillate; too small: never converge in the budget) and why concavity of the logistic regression LL means we reach the *global* optimum. Then write out the exact triple-nested pseudocode from lecture — repeat many times / zero the gradient accumulator / for each training example / for each parameter — and be explicit about what gets reset when and where the parameter update sits relative to the loop over examples. Contrast batch gradient ascent with stochastic gradient ascent."

Test me: > "Give me a two-datapoint, two-feature dataset, initialize $\theta = \mathbf{0}$, and walk me through *one* full step of gradient ascent by hand: predictions, log-likelihood, gradient, update, new predictions, new

log-likelihood. Check that I got the log-likelihood to increase. Then show me buggy training pseudocode (e.g. the accumulator reset in the wrong place, or the update inside the example loop when it shouldn't be) and make me find the bug."

Wrap-up prompt (after all six concepts)

> "Give me one multi-part CS109-style problem in which I must (1) state the logistic regression assumption and compute a prediction from a given θ and x, (2) write the decision boundary and say whether a given point is classified 1 or 0, (3) write the log-likelihood of a small labelled dataset and derive $\frac{\partial LL}{\partial \theta_j}$ from scratch, and (4) carry out one step of gradient ascent by hand and verify the log-likelihood increased. Let me solve every part, then grade each part, tell me which concept it tested, and tell me the single concept I should review most."

Note: everything here is maximum likelihood estimation — the exact recipe from last lecture, applied to a model whose parameter is itself a function of the input. Deep learning (in two lectures) is this same derivation, stacked. If the chain rule felt shaky today, shore it up now; it is the entire content of backpropagation.

Bring any sticking points to lecture; the TAs and instructor will help you work through them on the worksheet.