CS109 Lecture 21: LLM Learning Guide

Comparing Classifiers

Summer 2026

Lecture 21: LLM Learning Guide

Before lecture: read the Lecture 21 slides (comparing classifiers). 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: Brute force Bayes and the curse of dimensionality

Learn: > "Explain the 'brute force Bayes' classifier: predict $\arg\max_y P(Y = y \mid X = x)$ by storing a full conditional probability table over every assignment of the features. Work the table concretely for $m = 2$ and $m = 3$ binary features, then derive that the number of parameters is $O(2^m)$. Explain why this is hopeless for $m = 100$ — and be precise that the fatal problem is not memory but *data*: almost every cell in the table would have zero training examples in it. Connect this to the phrase 'curse of dimensionality.' "

Test me: > "Give me a dataset description with some number of binary features and make me compute how many parameters brute force Bayes needs, then ask me how many training examples I would need for the estimates to mean anything. Then ask me to explain, without formulas, why a table with more cells than datapoints is useless."

Concept 2: The Naive Bayes assumption

Learn: > "State the Naive Bayes assumption — features are conditionally independent given the class label, so $P(x_1, \ldots, x_m \mid y) = \prod_j P(x_j \mid y)$ — and show how it collapses the parameter count from $O(2^m)$ to $O(m)$. Walk through training (just counting) and prediction (multiply the per-feature conditionals times the class prior, compare across classes). Then be honest about the assumption: it is almost always false, since real features are correlated; explain why the classifier often works well anyway. Finally, explain why we use a Beta prior (Laplace or otherwise) on each conditional probability instead of raw MLE counts, and what specifically goes wrong when an MLE count is zero."

Test me: > "Give me a tiny Naive Bayes training set (say 3 binary features, 10 datapoints), make me estimate all the parameters by counting, then make me classify a new datapoint by hand. Then engineer the

dataset so one conditional count is zero, and make me explain what breaks and how a Laplace prior fixes it. Then ask me: is Naive Bayes a linear classifier?"

Concept 3: Train/test splits, baselines, and overfitting

Learn: > "Explain the train/test paradigm: why measuring accuracy on the data you trained on is a buggy metric, what a train/test split is, and what the split is protecting you from. Define overfitting and explain how to spot it from a table of train and test accuracies. Then explain why every results table needs a *baseline* (e.g. always predict the majority class), using an example with a 90/10 class imbalance where 89% accuracy is worse than useless. Also mention the validation set: why you need a third split if you are using the test set to choose between models."

Test me: > "Show me a table of models with train and test accuracies plus a baseline, and quiz me: which model overfits and how do I know, which model would I actually deploy, and is the best model meaningfully better than the baseline. Then give me a scenario with imbalanced classes and make me compute the majority-class baseline accuracy before I'm allowed to be impressed by any model."

Concept 4: Calibration

Learn: > "Explain what it means for a classifier's probabilities to be *calibrated*: of all the datapoints where the model said 0.8, about 80% should actually have label 1. Describe the procedure for checking this — bucket the test datapoints by predicted probability, then compute the observed fraction of positives in each bucket — and describe what a calibration curve looks like when the model is calibrated, overconfident, and underconfident. Then make the key distinction: calibration and accuracy are different properties. Give me an example of a model that is perfectly calibrated but has no discriminative power, and an example of a model that is accurate but badly calibrated. Finally, explain Platt scaling as a fix, and say why it must be fit on held-out data."

Test me: > "Give me a table of probability buckets with counts of $Y = 0$ and $Y = 1$ in each, and make me compute the observed fraction per bucket, say which buckets are miscalibrated and in which direction, and compute the accuracy within one bucket. Then make me use a Binomial to quantify how surprising the worst bucket is under the assumption that the model was calibrated, and ask me how I would put error bars on an observed fraction."

Concept 5: Precision, recall, and choosing a metric

Learn: > "Define the confusion matrix (TP, FP, FN, TN) and then accuracy, precision, and recall in terms of its entries, with a sentence of plain-English meaning for each ('of the ones I flagged, how many were real' vs. 'of the real ones, how many did I catch'). Explain the precision-recall tradeoff as you sweep the classification threshold, and what the precision-recall curve shows. Then work through several application domains — fraud detection, spam filtering, cancer screening, criminal sentencing — and reason about which metric matters based on the *cost* of a false positive versus a false negative in each. Mention F_1 as a summary and when it is and isn't appropriate."

Test me: > "Give me a confusion matrix and make me compute accuracy, precision, recall, and F_1. Then make me compute what the trivial 'always predict the majority class' baseline scores on each of those metrics, and explain why accuracy alone would have misled me. Then describe an application and make me argue for a metric and a threshold, and push back on my reasoning if I ignore the cost asymmetry."

Concept 6: Algorithmic fairness

Learn: > "Explain the two philosophies of fairness from lecture. **Procedural fairness** concerns the decision-making *process* and asks whether the algorithm relies on unfair features; 'fairness through unawareness' (just

delete the protected attribute) is its simplest form. **Distributive fairness** concerns *outcomes*. Define the two distributive criteria formally, using D for the protected demographic, G for the model's guess, and T for the truth: **parity**, $P(G = 1 \mid D = 1) = P(G = 1 \mid D = 0)$, and **calibration**, $P(T = 1 \mid G = g, D = 1) = P(T = 1 \mid G = g, D = 0)$, plus the relaxed version with tolerance ε and the US legal 80% rule for disparate impact. Then explain why fairness through unawareness usually fails — sensitive attributes are redundantly encoded across many correlated features — citing the Facebook 'Special Ad Audience' study. Finally explain the COMPAS case and why parity and calibration can conflict."

Test me: > "Give me a table of predictions and true outcomes broken down by a protected group, and make me compute both the parity gap and the calibration gap, check them against the 80% rule and an $\varepsilon = 0.2$ tolerance, and state which criterion the model satisfies. Then ask me the hard question: if a model passes one and fails the other, what should I actually say about it? Push back if I claim there is one correct definition of fairness."

Wrap-up prompt (after all six concepts)

> "Give me one multi-part CS109-style problem in which I must (1) compute the parameter count for brute force Bayes on a described dataset and explain why the Naive Bayes assumption is needed, (2) estimate a Naive Bayes parameter using a Beta prior and explain the effect of the prior, (3) read a train/test results table and identify overfitting and the relevance of the baseline, (4) check calibration from bucketed counts and say whether the model is over- or underconfident, and (5) compute precision and recall from a confusion matrix and argue for the right metric given the application. 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: today is the lecture where probability turns into engineering judgment. The math in each individual piece is light — fractions, a Binomial, a Beta posterior — but the reasoning about which number to trust is the part that shows up in real work, and on the exam.

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