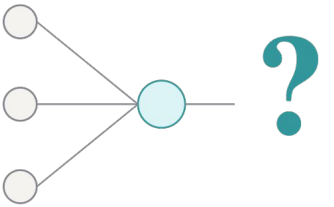
LOGISTIC REGRESSION 5 MIN

Testing Logistic Regression



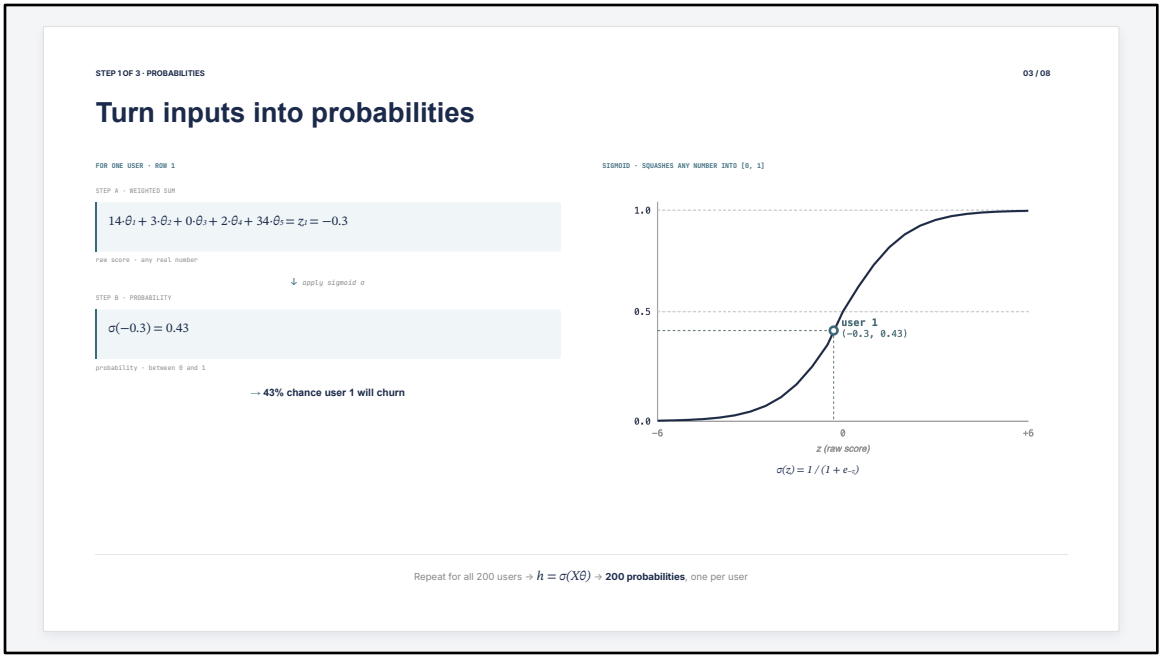
From a trained model to a number that tells you how well it works — and what to do next.

You've trained your logistic regression model. So how do you know it actually works — not just on the data it learned from, but on data it's never seen before? That's the whole point of testing. In this video, we'll walk through how to evaluate your model and what to do with the result.

Imagine a model that predicts whether a subscriber will cancel — or churn. To test it, you set aside data your model has never seen — what we'll call the held-out set.

Each row in this table is one user, described by five numbers like age and weekly logins. With 200 users, that's a matrix of 200 rows by 5 columns.

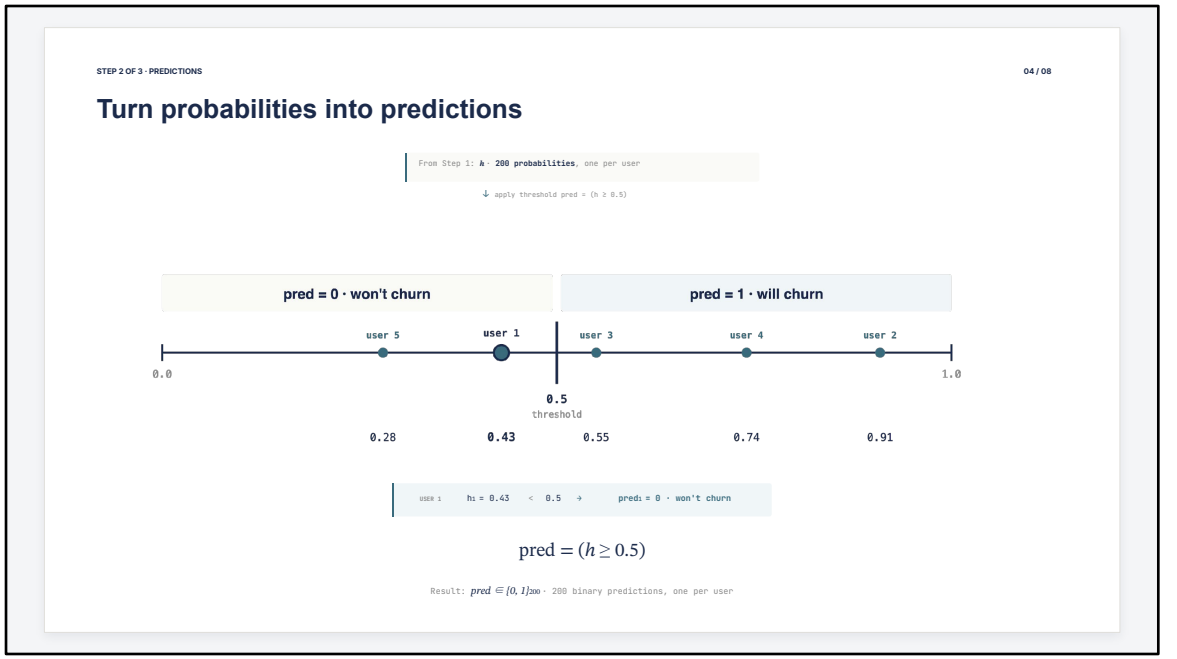
To test, you need three things: the held-out features X, the true labels y, and theta — the learned weights. We use new data to see how well our model generalizes. The process has three pieces: turn inputs into probabilities, threshold them into yes-or-no predictions, and check your accuracy.



First, we turn those inputs into probabilities. We multiply a user's five numbers by their weights and sum them to get a raw score.

User 1 might get a negative zero point three. But raw scores can be any number, and probabilities must run from zero to one.

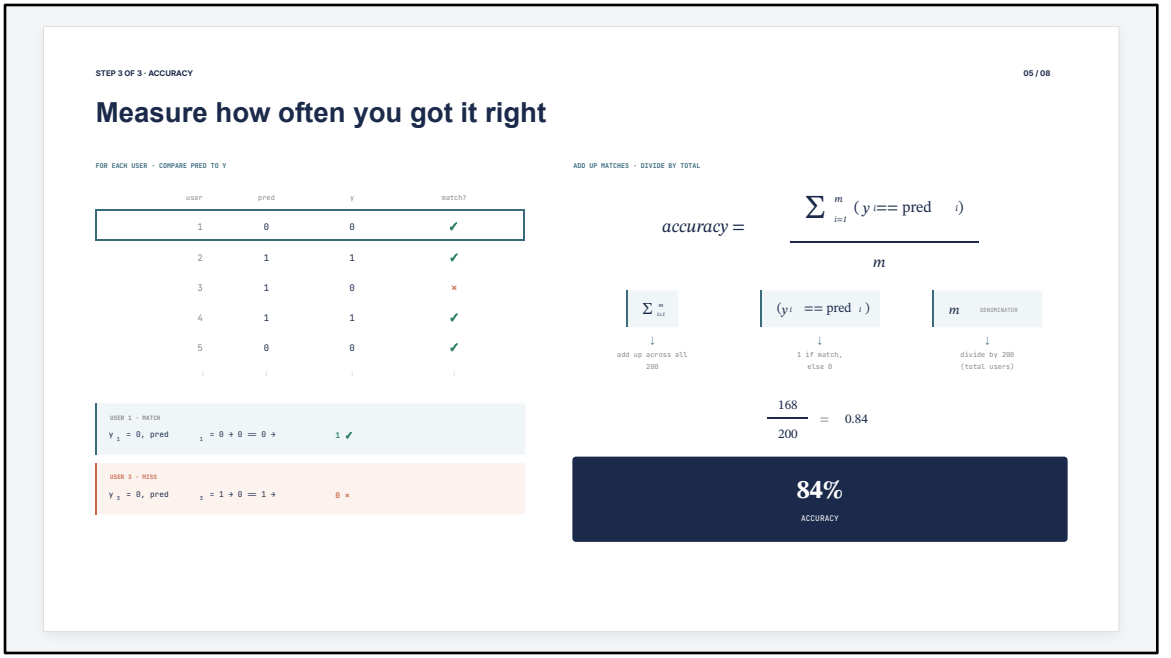
The sigmoid function fixes this by squashing any number into that zero-to-one range. Big positives end up close to one, big negatives close to zero. So sigmoid of negative zero point three is zero point four three — a forty-three percent chance of churning. Doing this for all 200 users gives us h, our list of probabilities.



Now that we have probabilities, we face a new problem. Your business can't actually act on a probability. It needs a clear answer — will they churn, or not?

So we set a threshold, usually zero point five. If a probability is zero point five or higher, we predict one — they'll churn. Anything below, we predict zero — they won't.

What about someone right at the edge — zero point five-one, or zero point four-nine? That's exactly why we need the threshold. It forces a clean decision. The result is pred — a list of 200 zeros and ones, one for each user.

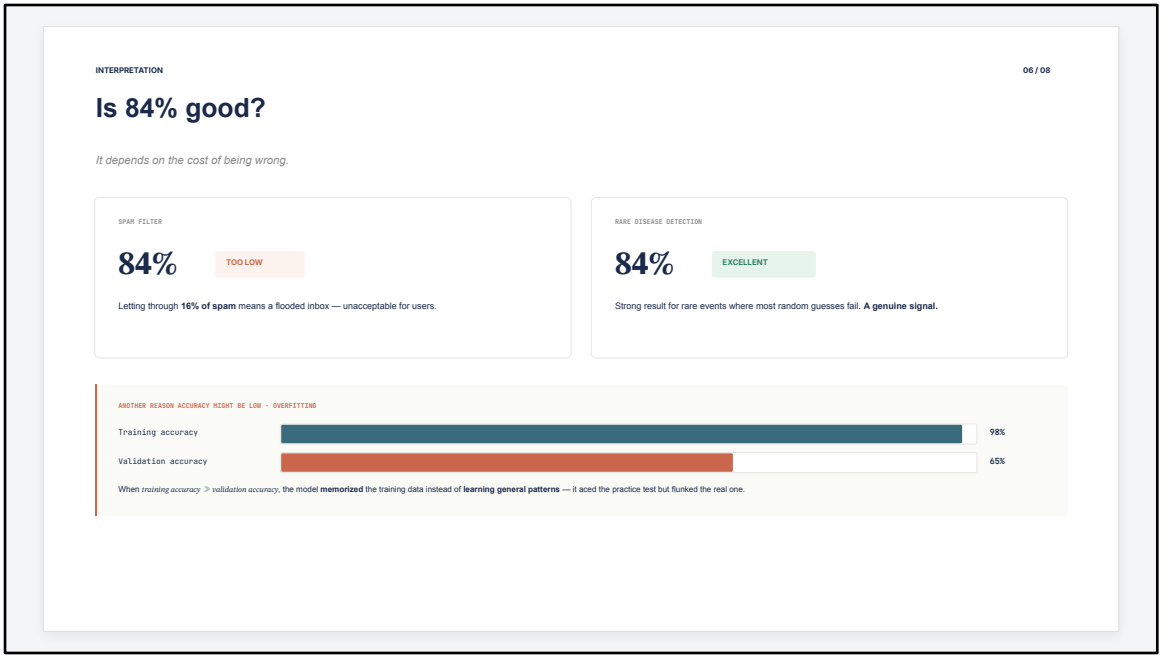


With predictions in hand, we can finally check how well our model did. For each user, compare your prediction to the true label. Count the matches. Divide by the total. That gives you accuracy — a single number that summarizes all 200 comparisons.

Accuracy runs from zero to one. One — or one-hundred percent — means you got every prediction right. Zero means you got none.

Let's say you got 168 out of 200 right. That's 168 divided by 200, or zero point eight-four — eighty-four percent accuracy.

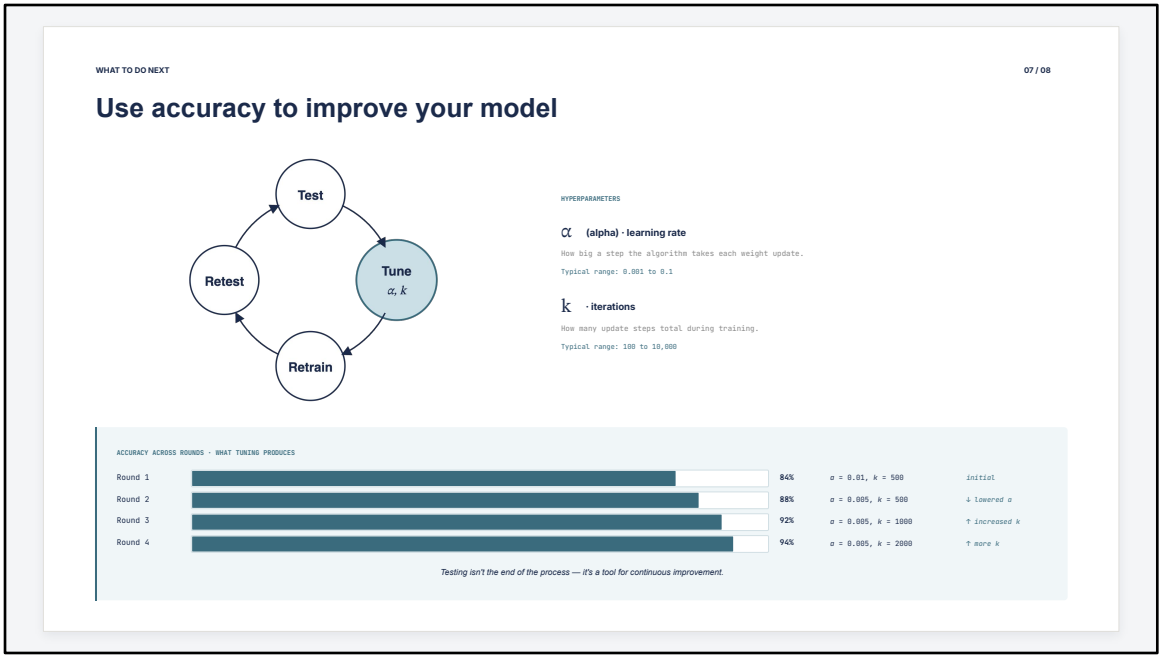
So… is that good? **[pause for thought]**



It depends. It depends on the cost of being wrong.

For predicting a rare disease, eighty-four percent might be excellent. For a spam filter, it's not nearly enough — sixteen percent of spam landing in your inbox every day is unacceptable.

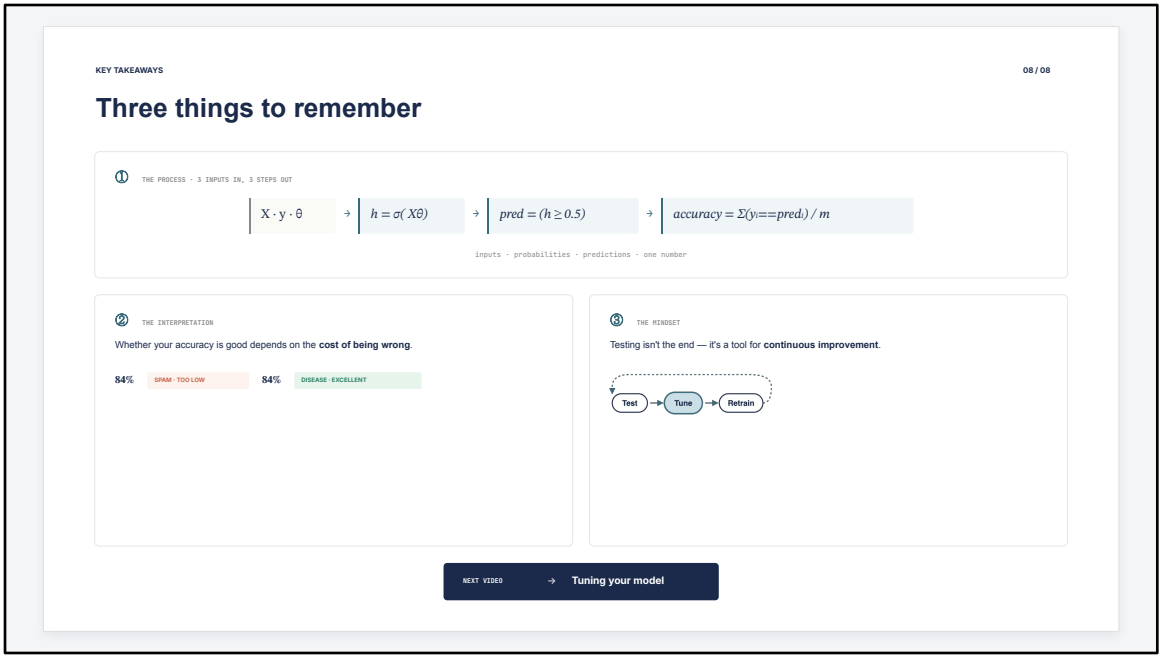
There's another reason accuracy might be low: overfitting. That's when your model memorized the training data instead of learning the actual patterns. How do you spot it? Training accuracy is much higher than held-out accuracy. In other words, the model aced the practice test but flunked the real one. We'll explore how to handle overfitting another time.



Accuracy is your guide for what to change next. You have two main knobs to turn, or **hyperparameters**: alpha, the learning rate, and k, the number of iterations. Adjust these, and your accuracy changes.

But there's an important caveat. The moment you use this held-out data to tune your model, it becomes a **validation set**. To know your model's real-world accuracy, you need one final, completely untouched **test set** — checked only after all tuning is done.

Testing isn't the end of the process — it's a tool for continuous improvement.



So, the big picture. Testing tells you whether your model truly generalizes — not just memorizes. Whether the accuracy is good depends on the cost of being wrong. And testing isn't the finish line — it's where tuning begins.

In the next video, we'll start with the learning rate — how a small change in alpha can mean the difference between a model that quickly converges and one that wanders for thousands of iterations.

See you there.