ELEC3544 · Fall 2026

Data Science with Foundation Models

Yang Liu

The University of Hong Kong



Classification - Predicting Categories

Where We Are

Module 2 builds the core machine learning toolkit. Each lecture tackles one problem type, applies the same evaluation discipline.

Lecture 5

Regression

Predicting Quantities

Method: Linear regression

Evaluation: R^2, RMSE, residual analysis

CTA pattern: Verification through computation

Lecture 6 · Today

Classification

Predicting Categories

Method: Logistic regression

Evaluation: Confusion matrix, ROC, AUC

CTA pattern: Verification + Iterative refinement

Lecture 7 · Next

Unsupervised

Finding Structure Without Labels

Method: Clustering, embeddings

Evaluation: Silhouette, inertia

CTA pattern: Exploratory

Today's Journey: Three Questions That Drive the Complete Classification Workflow

Session 1

Can we predict a binary outcome with linear regression — and if not, what model do we need?

We fit linear regression to a binary target live with the CTA, discover where it breaks, and build logistic regression as the principled fix.

Session 2

How do we turn predicted probabilities into decisions — and how do we measure whether those decisions are good?

We construct the confusion matrix, derive precision and recall, and expose a classifier with 73.5% accuracy while being completely useless.

Session 3

Is there a way to evaluate a classifier without committing to a threshold — and when should we use a different model?

We build the ROC curve and AUC, verify the probabilistic meaning of AUC empirically, then compare logistic regression to a decision tree.

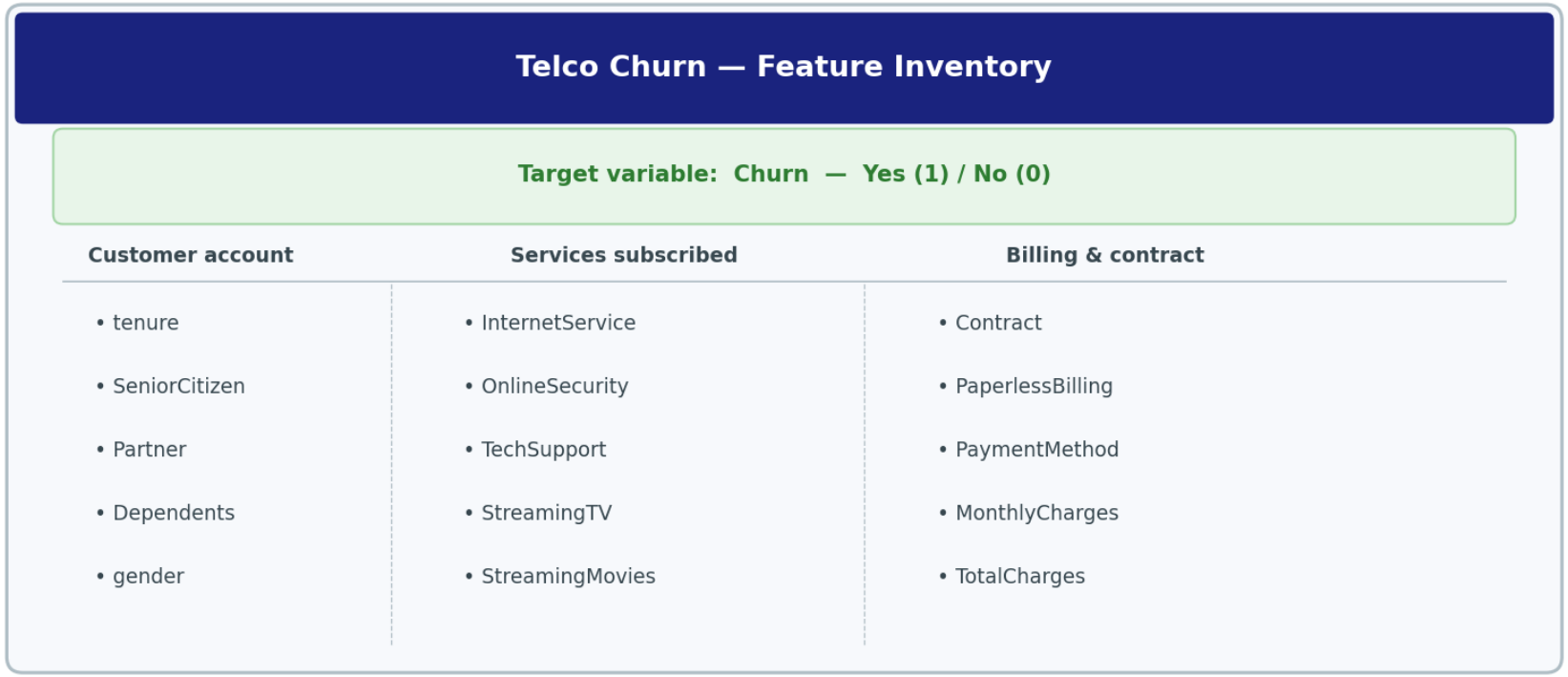
Example: Telco Churn — Ready for Classification

Dataset Overview

Observations	**7,043 rows**
Features	**19 raw predictors**
Target	**Churn (yes / no)**
Problem type	**Binary classification**
Data source	**Cleaned in Lecture 4**

Callback to L4:

Missing values handled, categorical variables encoded. Ready to use.



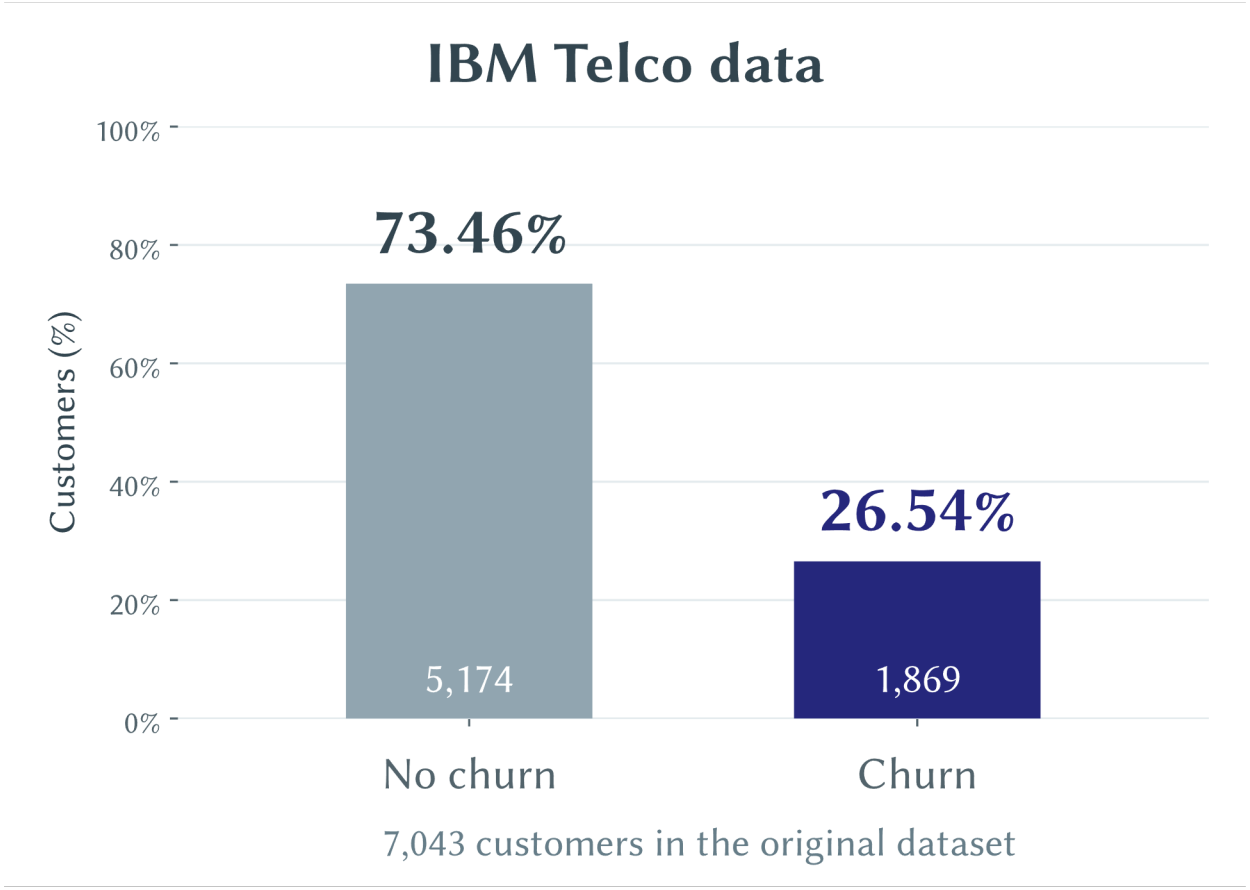
The Churn Dataset Has a Built-In Challenge: About 73.5% Do Not Churn

73.5%

No Churn ($y = 0$, negative class)

26.5%

Churn ($y = 1$, positive class)



Its significance becomes clear in Session 2, when we ask: can a classifier that never predicts churn still achieve strong accuracy on the standard metric?

Why Classification Needs Its Own Model

Session question:

Can we predict a binary outcome with linear regression — and if not, what model do we need?

Concepts covered:

Linear regression failure · Sigmoid function · Logistic regression · Log-odds

[Opening Question] Can We Just Use Linear Regression to Predict Who Will Churn?

Why not just use what we already know?

> *"We have linear regression. The churn column is just zeros and ones. Why not simply apply what we already know?" — Consider this before we try it live with the CTA.*

CTA Demo 1: Fitting Linear Regression to a Binary Target — Let's See What Breaks

🤖 CTA Demo 1: Fitting Linear Regression to a Binary Target

Turn 1: Ask the CTA to fit `LinearRegression` to the churn target using `tenure` alone, plot the fit over 0–90 months, mark the observed 0–72 month range, and report where extrapolated predictions leave [0, 1].

Turn 2: Ask: "What does a prediction of 1.3 or −0.2 mean as a probability?" Then ask the CTA to fit `LogisticRegression` to the same data, plot the sigmoid curve, and confirm all predictions stay within (0, 1).

Takeaway: Linear predictions are unbounded and can leave the valid probability range. The sigmoid enforces (0, 1); calibration still needs checking.

Linear Regression Can Predict Outside [0,1] — Invalid as Probabilities

1. Output range

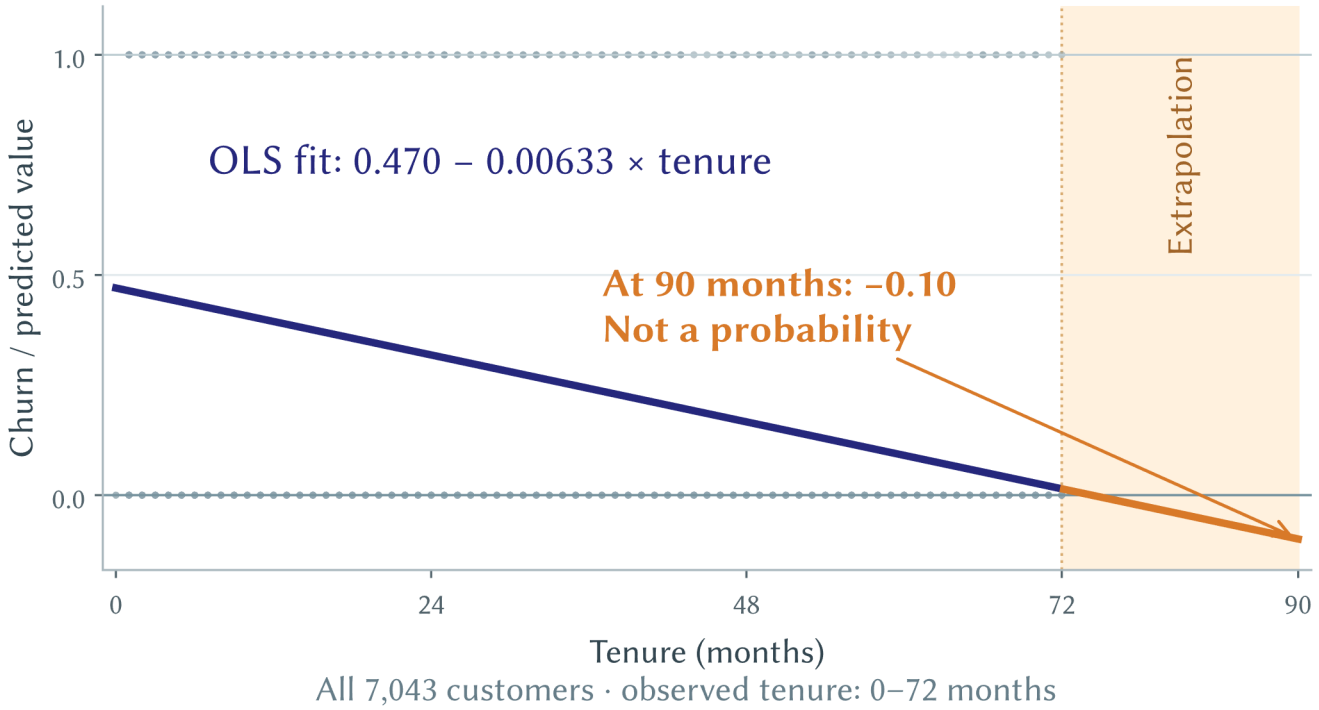
Linear regression can predict values outside [0, 1]. A prediction of −0.2 or 1.3 cannot be a probability — probabilities must lie in [0, 1] by definition.

2. Calibration

Being inside [0, 1] does not guarantee calibration. Check whether predicted probabilities match observed frequencies on held-out data.

Solution:

Logistic regression wraps the linear predictor in a function that maps any real input to a value in (0, 1).



The Sigmoid Function Maps Any Finite Input Between 0 and 1

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

Three key properties:

Range:

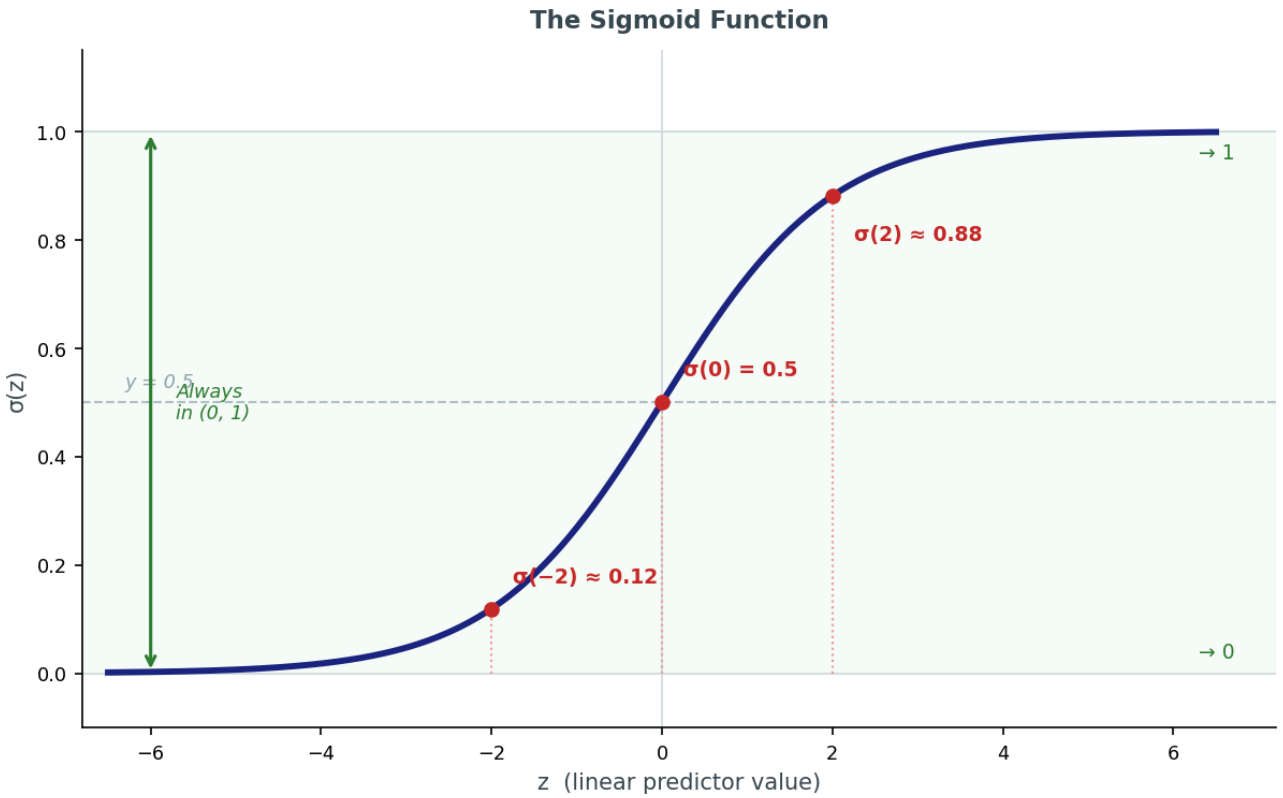
Output is always in (0, 1). This gives a valid probability range, not a guarantee of calibration.

Symmetry:

$\sigma(0) = 0.5$. Positive z gives probability > 0.5; negative z gives probability < 0.5.

Asymptotic:

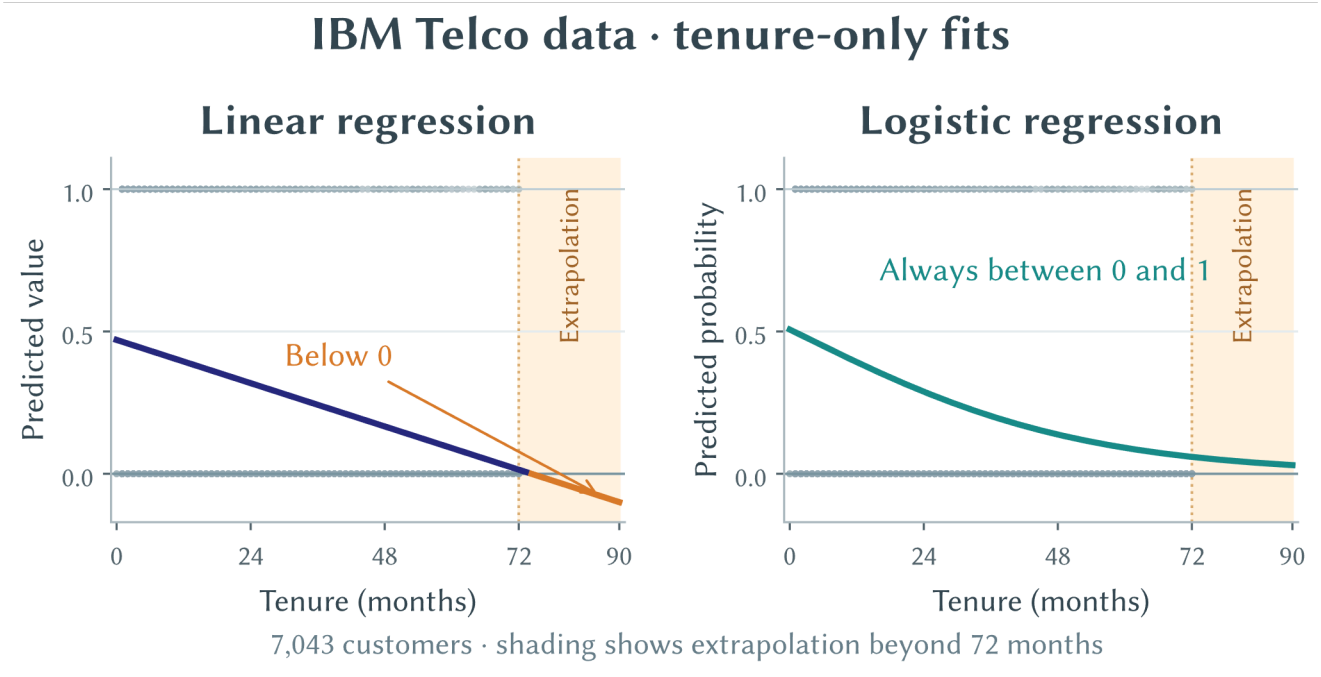
As $z \longrightarrow +\infty$, $\sigma(z) \longrightarrow 1$. As $z \longrightarrow -\infty$, $\sigma(z) \longrightarrow 0$. Never exactly reaches 0 or 1.



Logistic Regression Wraps a Linear Model Inside a Sigmoid — That Is the Complete Idea

$$P(y = 1 \mid x) = \sigma(\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p)$$

Key idea: the model outputs a probability — a number in (0, 1) — not a class label. What to do with that probability is a separate decision made later.



Note: β coefficients minimize negative log-likelihood, usually with a regularization penalty

The Logistic Regression Equation: Two Equivalent Forms, Two Ways to Think About It

Probability form

Operationally useful

$$P(y = 1 \mid x) = \sigma(\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p)$$

Directly produces a value in (0, 1). Use this form when predicting and scoring observations.

Log-odds form (logit)

Interpretively useful

$$\log\left[\frac{P}{1-P}\right] = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$$

The left side is the log-odds. The right side is a linear model. Coefficients are interpretable in log-odds space — this is what makes them useful for domain reasoning.

Odds = P / (1 – P). If P = 0.75, odds = 0.75 / 0.25 = 3-to-1 in favour of churn. Log-odds = log(3) ≈ 1.10.

The two forms are mathematically equivalent. The log-odds form is what makes coefficients interpretable in plain language. In the next section, we fit the model, run the code in Colab, and the output will include a coefficient table — one number per feature. The question we then face is what those numbers actually mean for a business decision.

A Coefficient of –0.89 for Contract Type — What Does That Mean?

$$\beta = -0.89$$

long-term contract (1) vs. month-to-month (0)

> *Consider an illustrative coefficient of –0.89 for long-term contract (1) versus month-to-month (0). What does it mean? Does it justify offering long-term contracts to everyone at risk of churning?*

We can ask the CTA to translate this number — and then probe whether acting on its answer is safe.

CTA Demo 2: Interpreting the Fitted Churn Model in Plain Business Language

🤖 CTA Demo 2: Interpreting the Fitted Churn Model in Plain Business Language

Setup: Use the illustrative coefficients shown here, or coefficients from your own fitted model. State each feature's unit and reference category before interpreting it.

Turn 1: Ask the CTA to convert the top four logistic regression coefficients to odds ratios using `np.exp(coef_)` and write a plain-language interpretation of each for a business audience.

Turn 2: Ask whether the negative coefficient for contract type means the company should immediately offer long-term contracts to all at-risk customers. The CTA should flag the causal inference limitation — correlation in historical data does not guarantee that an intervention on the variable would produce the same effect.

Takeaway: Check both the mathematical translation and the causal caveat: an association does not establish that changing a feature will change churn. This is the first explicit human-judgment moment of the lecture.

The Translation We Need: Log-Odds → Odds Ratio → Plain Language

$$\text{Odds Ratio} = e^{\beta_j}$$

Step 1 — Start with the log-odds coefficient

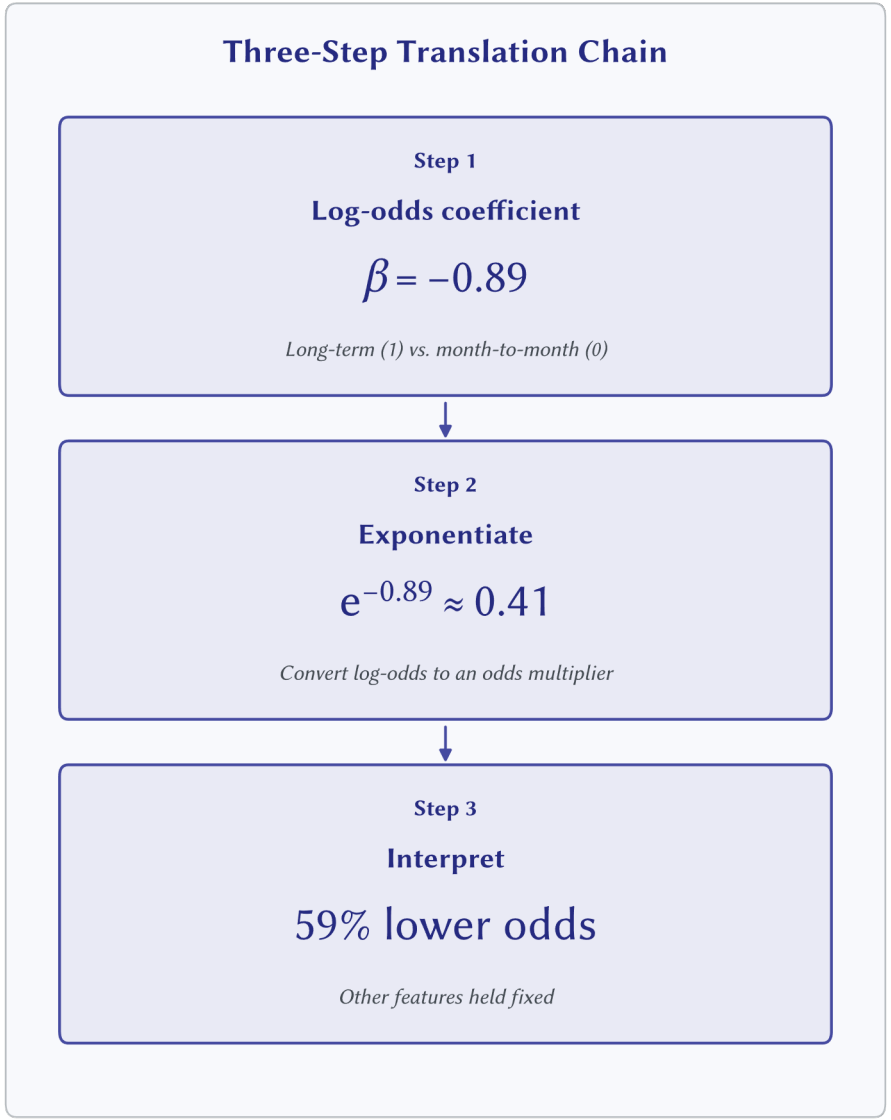
Illustrative $\beta = -0.89$ for long-term contract (1) versus month-to-month (0). The coefficient is a change in log-odds.

Step 2 — Exponentiate to get the odds ratio

$e^{-0.89} \approx 0.41$. The odds ratio tells us by what factor the odds of churning are multiplied for each unit increase in the feature.

Step 3 — Translate to plain language

Long-term contract customers have odds of churning that are approximately 59% lower than month-to-month customers, holding all other features constant.



Odds ratio < 1: lower modelled odds per unit increase. > 1: higher modelled odds. = 1: no modelled change.
These are associations, not causal effects.

Illustrative Coefficients: How Does Each Feature Multiply the Odds?

Feature	β (log-odds)	e^{β} (odds ratio)	Plain-language interpretation
Long-term contract (1) vs. month-to-month (0)	**−0.89**	**0.41**	~59% lower odds vs. month-to-month
Tenure (years)	**−0.46**	**0.63**	Per extra year: odds × 0.63
Monthly charges ($10 units)	**+0.38**	**1.46**	Per extra $10: odds × 1.46
No tech support	**+0.31**	**1.36**	No support (1) vs. support (0): odds × 1.36
Paperless billing	**+0.22**	**1.25**	Paperless (1) vs. not (0): odds × 1.25

Sign convention (holding other features fixed):

$\beta < 0 \longrightarrow$ odds ratio < 1; $\beta > 0 \longrightarrow$ odds ratio > 1; $\beta = 0 \longrightarrow$ odds ratio = 1. These describe modelled associations.

Predicting and Explaining Are Different Goals — Coefficients Are Not Causal Claims

Key Insight

A logistic regression coefficient tells us: in the historical data, customers with feature *X* had higher (or lower) odds of churning than customers without it, after controlling for the other features in the model.

It does not tell us: if we change feature *X* for a customer, their churn probability will change by that amount. The coefficient captures association, not causation.

Concretely: "customers on long-term contracts churn less" does not mean "moving customers to long-term contracts will reduce churn." The historical pattern may reflect self-selection — loyal customers choose long contracts — rather than a treatment effect.

The model CAN:

Score customers by predicted churn probability

Describe feature associations with stated units

Guide where to investigate further

The model CANNOT:

Prove that offering a long-term contract will reduce a customer's churn risk

Replace a controlled experiment for causal questions

Specify which interventions are cost-effective

Bridge: "Let us now consolidate Session 1 before we turn to the harder question: what do we actually do with these probabilities?"

Session 1 Summary: Logistic Regression Gives Us Probability Estimates

1. Linear regression can leave the valid probability range

Its predictions are unbounded and may fall outside [0, 1]. Logistic regression enforces a valid range.

2. The sigmoid ensures a valid probability range

$\sigma(z) = 1/(1+e^{-z})$ maps any real input into (0, 1). Calibration still needs checking on held-out data.

3. Coefficients live in log-odds space

To interpret them, exponentiate: e^{β} gives the odds ratio. The translation chain — log-odds ⟶ odds ratio ⟶ plain language — makes them business-readable.

4. Coefficients are associations, not causal claims

The model can score and rank customers. It cannot prescribe interventions. Acting on predictions as if they were causal requires domain judgment the model cannot supply.

Bridge: "We now have a model that outputs probabilities between 0 and 1 for every customer. But a probability is not a decision. The next question is: how do we turn these numbers into actions — and how do we know whether those actions are good?"

Mid-term Exam (Nov. 21 – in-class 1.5~2h)

See Moodle for the confirmed schedule.

Check the current announcement for the venue, coverage, duration, and permitted materials.

From Probabilities to Decisions: Building the Evaluation Toolkit

Session question:

We have a predicted probability for every customer — how do we decide who gets flagged, and how do we measure whether those decisions are good?

Concepts covered:

Confusion matrix · Precision · Recall · Threshold · Precision-recall curve · F1 score

[Opening Question] We Have a Predicted Probability for Every Customer — How Do We Decide Who Gets Flagged?

The model outputs a number between 0 and 1 for every customer. That is not yet a decision. To act, we need a rule.

Flag as churner if P(churn) ≥ ?

0.3

Flag aggressively Catch more churners but also flag loyal customers

0.5

Default choice Seems fair — but is it the right cut-off here?

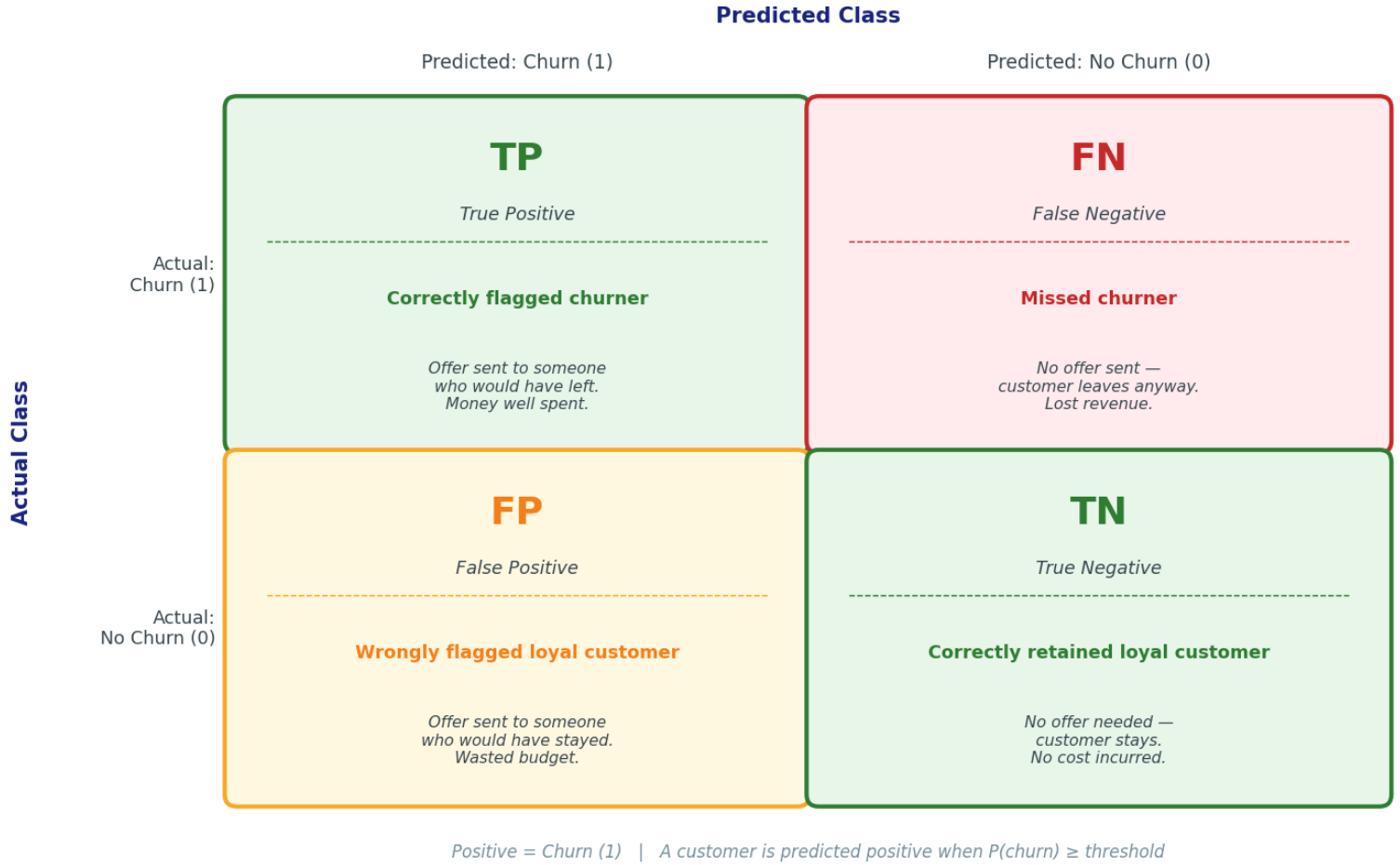
0.7

Flag conservatively Higher confidence per flag but miss many real churners

> *The threshold is a choice, not a given. The right value depends on the cost of each type of error.*

The Confusion Matrix: Four Outcomes, Four Consequences, One Complete Picture of Model Performance

A threshold converts predicted probabilities into class labels. Those labels fall into exactly four outcome types.



Threshold (θ):

Predict churn (1) if P(churn | x) ≥ θ
Predict no churn (0) otherwise

Default $\theta = 0.5$, but this is a choice — not a statistical requirement.

Positive class = churn = 1. True/False refers to whether the prediction was correct.

Precision: When We Flag a Churner, How Often Are We Right?

$$\text{Precision} = TP / (\underbrace{TP + FP}_{\text{All predicted positives}})$$

Of all the customers we flag as likely churners, what fraction actually churn?

High precision

Most customers we flag are real churners. Retention budget is spent efficiently — few discounts wasted on loyal customers.

Low precision

Many flagged customers are loyal. The retention campaign sends unnecessary discounts, wasting budget on customers who were not leaving.

TP = true positives (correctly flagged churners). FP = false positives (loyal customers incorrectly flagged). Both come from the confusion matrix.

Recall: Of All Customers Who Will Churn, How Many Did We Catch?

$$\text{Recall} = TP / (\underbrace{TP + FN}_{\text{All actual positives}})$$

Of all the customers who actually churn, what fraction did our model catch?

High recall

Most customers who would have churned are flagged. Few churners slip through — lost revenue is minimized.

Low recall

Many real churners are missed. Those customers leave without intervention — each represents lost subscription revenue.

TP = true positives (churners we caught). FN = false negatives (churners we missed). Note: Recall = TPR — the same quantity used in the ROC curve (Session 3).

An Illustrative Worked Example: Computing Precision and Recall

Illustrative cohort: 2,000 customers, including 400 churners. At threshold = 0.5, these four counts show how the metrics work:

	Predicted: Churn	Predicted: No Churn
Actual: Churn	**TP = 180**	**FN = 220**
Actual: No Churn	**FP = 120**	**TN = 1,480**

Precision:

TP / (TP + FP) = 180 / (180 + 120) = 180 / 300 = 0.60

60% of customers we flag as churners actually churn.

Recall:

TP / (TP + FN) = 180 / (180 + 220) = 180 / 400 = 0.45

We catch 45% of all customers who actually churn.

> *At threshold = 0.5 we catch fewer than half of all churners. Whether that is acceptable depends on the cost of missing them.*

CTA Demo 3: Thresholds and Hypothetical Business Utility

Illustrative payoffs:

Hypothetical payoff for TP: +$450; FP: −$50.

Hypothetical payoff for FN: −$500; TN: $0.

Total utility = $450 × TP − $50 × FP − $500 × FN

Illustrative worked example — 2,000 customers:

θ	Precision	Recall	Utility ($)
0.3	**0.48**	**0.71**	**+$54,400**
0.5	0.60	0.45	−$35,000
0.7	0.74	0.23	−$114,200

Among these three thresholds, 0.3 gives the highest utility. A full threshold sweep may find a better choice; 0.3 is not the global optimum.

CTA Demo 3: Thresholds and Hypothetical Business Utility

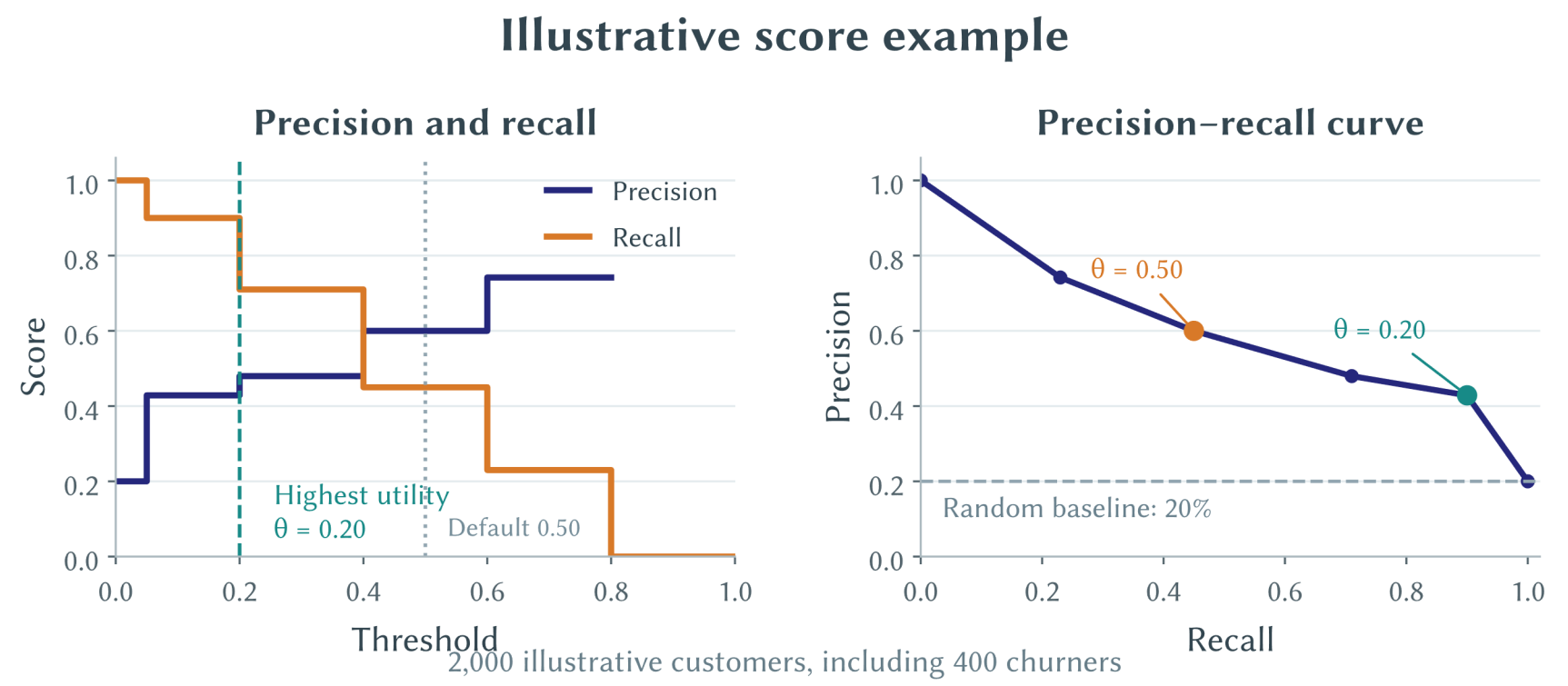
Turn 1: Provide these hypothetical payoffs. Ask the CTA to compute precision, recall and utility at thresholds 0.3, 0.5 and 0.7. Check the arithmetic against the illustrative table.

Turn 2: Ask the CTA to sweep all thresholds and mark the highest-utility point. For a real model, tune the threshold on validation data; evaluate once on the test set.

These hypothetical payoffs are business inputs, not quantities inferred from the churn data.

The Precision–Recall Curve Shows the Available Threshold Tradeoffs

Each point represents a threshold. Lowering the threshold cannot decrease recall; precision may rise or fall.



Top-right corner = perfect classifier

Precision = 1 and Recall = 1 simultaneously — catches every churner with zero false alarms.

Horizontal baseline = random classifier

Expected precision equals prevalence: 20% in this illustrative cohort. The standard Telco dataset has about 26.5% churn.

Threshold Selection Is a Business Decision — the Model Cannot Choose for You

Key Insight

The PR curve shows every possible precision-recall tradeoff. But it cannot tell you which point to choose.

Choosing a threshold means deciding: how costly is a false positive relative to a false negative? That is a question about business context — the cost of a wasted discount versus the value of a retained customer — not a question the model can answer.

The CTA can optimise against an explicit cost function once you provide one. It cannot derive the cost function from the data. Specifying what errors cost is irreducibly a human responsibility.

What the CTA can do:

Compute precision and recall at any threshold

Generate the full PR curve

Find the threshold that minimises a cost function you specify

What only a human can do:

Specify what a false positive costs in this business context

Specify what a false negative costs

Decide whether those tradeoffs are acceptable

F1 Score Penalises Models That Sacrifice One Side of the Precision-Recall Tradeoff

$$F_1 = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$$

F1 is the harmonic mean of precision and recall. Unlike the arithmetic mean, it is dominated by whichever value is smaller.

Arithmetic mean — misleading

Precision = 0.90, Recall = 0.10

Arithmetic mean = (0.90 + 0.10) / 2 = 0.50

Suggests a mediocre but acceptable classifier.

Harmonic mean — reveals the problem

Precision = 0.90, Recall = 0.10

F1 = 2 × (0.90 × 0.10) / (0.90 + 0.10) = 0.18

Reveals that the model almost never catches churners.

When to use F1:

F1 is appropriate when both precision and recall matter and you want a single number. It is especially useful for model comparison. When precision and recall have different costs, use the PR curve and cost-weighted optimization instead.

[Opening Question] We Have Precision, Recall, and F1 — Would Accuracy Alone Have Been Enough?

The standard Telco dataset has about 73.5% non-churners and 26.5% churners. What accuracy would an always-no-churn classifier achieve?

$$\text{Accuracy} = \frac{\text{Fraction of all predictions correct}}{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}$$

> **Hint: class distribution**
> Always predicting 'no churn' is correct for about 73.5% of the standard Telco dataset, but catches none of the churners.

CTA Demo 4: About 73.5% Accuracy Without Predicting Any Churn

🤖 CTA Demo 4: The Naïve Majority-Class Classifier

Setup: Ask the CTA to write code that fits `sklearn`'s `DummyClassifier(strategy='most_frequent')` on the Telco training set and evaluates it on the test set.

Turn 1 — Colab output: The dummy classifier achieves accuracy ≈ 0.735 (depending on the test split). Ask the CTA to also compute precision, recall, and F1 for both the dummy classifier and our fitted logistic regression. We run the comparison code in Colab.

Turn 2: Ask the CTA to explain why a model with about 73.5% accuracy can have recall = 0 and F1 = 0. The answer should make the class-imbalance mechanism explicit.

Takeaway: An always-majority classifier matches the majority-class proportion. Accuracy alone can conceal zero positive-class recall.

Accuracy Is a Warning Sign on Imbalanced Data — Not a Performance Guarantee

Key Insight

In our illustrative cohort with 80% non-churners, a classifier that always predicts 'no churn' achieves 80% accuracy. It never identifies a single churner.

Accuracy measures the fraction of correct predictions across all observations — but it weights the majority class proportionally to its size. When classes are heavily imbalanced, the majority class dominates the score.

Precision and recall break this down: they measure performance separately on positive predictions and the positive class. A model with recall = 0 cannot hide behind an inflated accuracy figure.

Illustrative comparison — same 2,000 customers as the worked matrix:

Metric	Naïve	Score model
Accuracy	**0.80** ⚠	0.83
Precision	—	0.60
Recall	**0.00** ✗	0.45
F1	**0.00** ✗	0.51

When a dataset is imbalanced, always report precision and recall alongside accuracy. If only accuracy is reported, ask about the class distribution first.

Metric Decision Guide: Which Evaluation Tool to Use and When

Each metric answers a different question. The right choice depends on the business context and class balance.

Metric	Question answered	Best for	Covered
Accuracy	Fraction of all predictions correct	When overall error rate matches the goal	S2 ✓
Precision	Of flagged positives, how many are real?	When FP is costly	S2 ✓
Recall	Of real positives, how many did we catch?	When FN is costly	S2 ✓
F1 Score	Harmonic mean of precision and recall	Single summary when both matter	S2 ✓
PR Curve	Full precision-recall tradeoff across thresholds	Imbalanced datasets	S2 ✓
ROC / AUC	Threshold-independent ranking quality	Comparing ranking quality	S3 ▸

Session 2 Summary: Decisions Need a Threshold; Curves Sweep All Thresholds

1. A probability is not a decision — the threshold is a choice

Converting predicted probabilities to class labels requires setting θ. That choice determines which customers are flagged and how many errors of each type occur.

2. Precision and recall separate the two types of error

Precision measures wasted effort (false positives). Recall measures missed opportunities (false negatives). Both are needed because accuracy hides which error type is dominating.

3. The PR curve shows every possible precision-recall tradeoff

Some thresholds can improve both precision and recall. The optimal threshold requires a cost function — and that function is a business judgment, not a statistic.

4. Accuracy alone can mislead

The illustrative all-negative baseline has 80% accuracy and zero recall. Read accuracy with class prevalence, precision, recall, and the cost of errors.

Threshold-Free Evaluation and Model Selection

Session question:

Is there a way to evaluate a classifier without committing to a threshold — and when should we use a different model entirely?

Concepts covered:

TPR · FPR · ROC curve · AUC · Probabilistic AUC interpretation · Decision tree · Model selection

[Opening Question] Can We Summarise Ranking Without Choosing a Threshold?

Precision, recall, and F1 are all computed at a specific threshold. Changing the threshold can change these values. Is there an evaluation method that captures model quality independently of that choice?

What if we swept every threshold at once?

The key idea:

Instead of fixing θ and computing one precision-recall pair, we compute the true positive rate and false positive rate at every possible threshold and trace the resulting curve. That curve — the ROC curve — summarises classifier quality across all thresholds simultaneously.

Two Rates Drive the ROC Curve: True Positive Rate and False Positive Rate

TPR = TP / (TP + FN) = Recall

FPR = FP / (FP + TN)

True Positive Rate (TPR)

Of all customers who actually churn, what fraction did our model flag?

TPR is identical to recall — the same computation under a different name. High TPR means we catch most churners.

False Positive Rate (FPR)

Of all customers who will NOT churn, what fraction did our model incorrectly flag?

FPR has no equivalent in the precision-recall framework. Low FPR means we rarely disturb loyal customers unnecessarily.

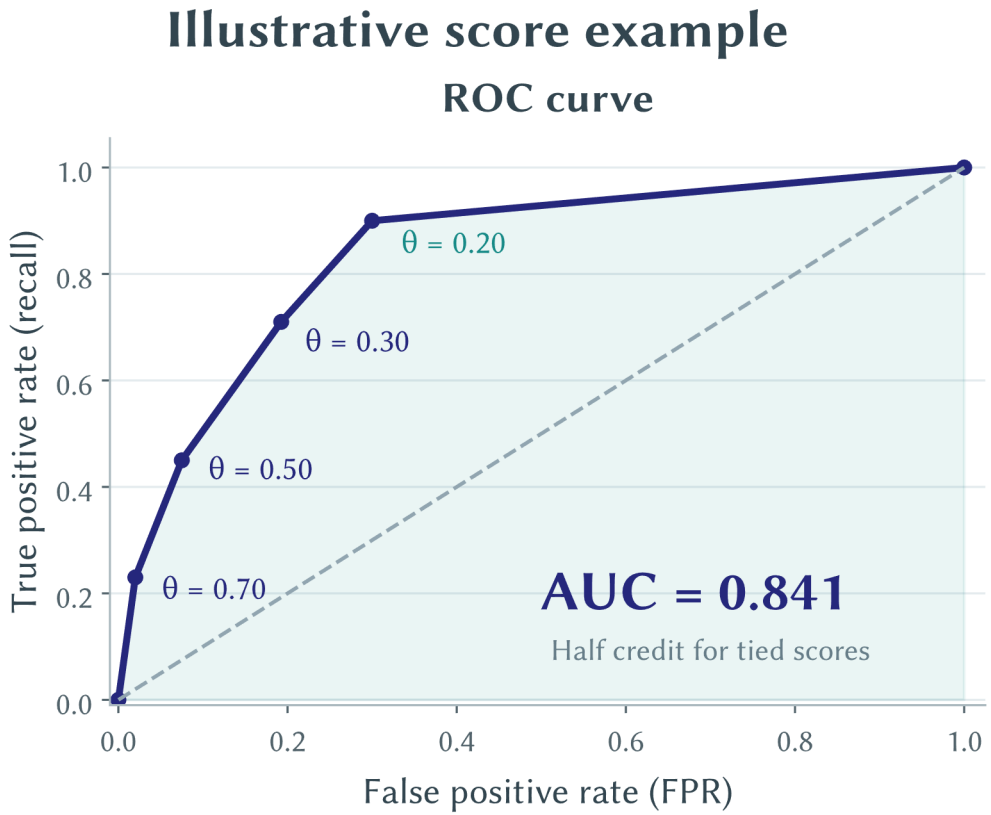
Illustrative model at $\theta = 0.5$ (from our confusion matrix):

TPR = 180 / (180 + 220) = 0.45 · FPR = 120 / (120 + 1480) = 0.075 · This is one point on the ROC curve.

TPR = Recall = Sensitivity. FPR = 1 – Specificity. These terms appear in medical and security literature.

The ROC Curve Sweeps Every Threshold and Plots TPR Against FPR

Each threshold gives one (FPR, TPR) pair. Sweeping thresholds traces the ROC curve; this figure uses the illustrative score set.



Top-left corner (0, 1) = Perfect classifier

TPR = 1 and FPR = 0 simultaneously — catches every churner with zero false alarms. A perfectly separated test set reaches this point.

Diagonal = random-ranking baseline

For random ranking, expected TPR equals FPR. Better ranking generally bends the curve toward the top-left.

Top-right corner (1, 1) = Everyone flagged

All positives are caught, but every negative is also flagged: TPR = FPR = 1 at this threshold.

AUC: The Area Under the ROC Curve Summarises Classifier Ranking Quality

AUC converts the entire ROC curve into a single number — but the number has a concrete probabilistic meaning.

$$\text{AUC} = P(s_+ > s_-) + \tfrac{1}{2} P(s_+ = s_-)$$

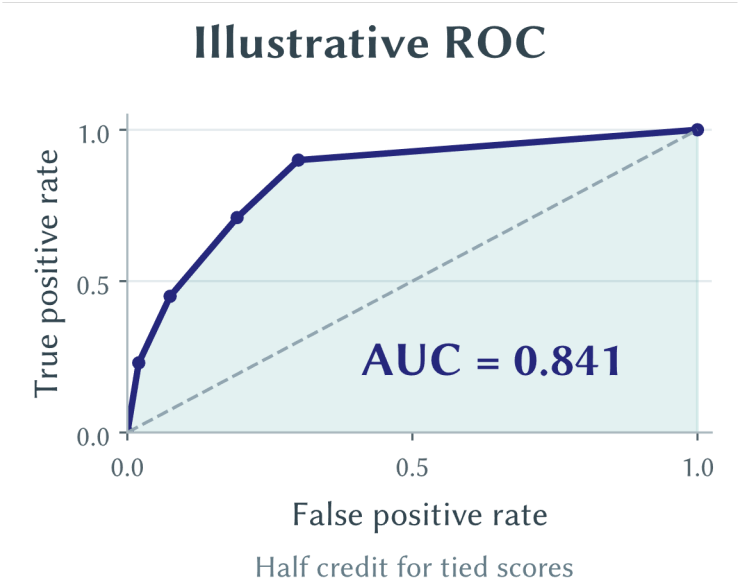
Compare a random churner and non-churner: a correct ranking earns 1, a tie ½, and an incorrect ranking 0.

What AUC = 0.84 means

In this illustrative score set, the mean ranking credit is about 0.84 across all churner/non-churner pairs, counting a tie as half a correct ranking.

AUC benchmarks

AUC = 1.0 ⟶ perfect ranking · AUC = 0.5 ⟶ chance-level average ranking credit · AUC = 0.0 ⟶ perfectly wrong (systematically ranks negatives above positives) Judge usefulness against relevant baselines, the operating region, error costs, and uncertainty.



Half credit for tied scores

This probabilistic interpretation is a specific claim — we can verify it computationally. We can ask the CTA to write code that estimates AUC by random pair sampling and check agreement with `sklearn` within sampling error.

CTA Demo 5 Setup: Can We Verify the Probabilistic Meaning of AUC Computationally?

AUC = $P(s_+ > s_-) + \frac{1}{2} P(s_+ = s_-)$. We can verify this by comparing positive/negative score pairs.

The verification challenge:

For the illustrative scores, `sklearn`'s `roc_auc_score()` gives about 0.84.

Sample one positive and one negative independently with replacement. Give 1 for a higher positive score, ½ for a tie, and 0 otherwise; average the credits. This estimate should be close to AUC, within sampling error.

This applies VERIFY's Execute and Reason steps to the meaning of a metric.

What we expect to see:

`sklearn` AUC: 0.84

Empirical pair-sampling AUC (10,000 random pairs): ≈ 0.84

The estimate should be close, within Monte Carlo sampling error.

VERIFY framework — Reason step: understanding what a metric measures before relying on it.

CTA Demo 5: Empirical Verification of AUC via Random Pair Sampling

🤖 CTA Demo 5: Verification Through Computation

Setup: Ask the CTA to write two code blocks: (1) compute AUC using `sklearn`'s `roc_auc_score()`, and (2) estimate AUC empirically by randomly sampling 10,000 churner/non-churner pairs and averaging 1 for a higher churner score, ½ for a tie, and 0 otherwise. We run both in Colab and compare the outputs.

Turn 1 — Illustrative scores: `sklearn` AUC ≈ 0.84; the pair-sampling estimate should be close, within sampling error. Ask the CTA to explain why they agree approximately and why more pairs reduces sampling variability.

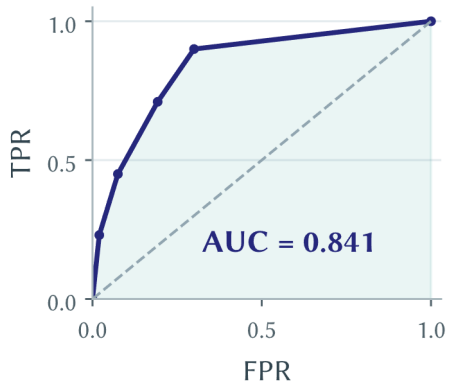
Turn 2: Ask the CTA what AUC = 0.84 means in the context of the churn retention campaign. It should explain average ranking credit, including ties, and why this alone does not establish campaign value.

Takeaway: The CTA performs the verification fluently. The key point is that metrics have meanings — and those meanings can be tested, not just trusted. This is the VERIFY framework applied to the output of a library function.

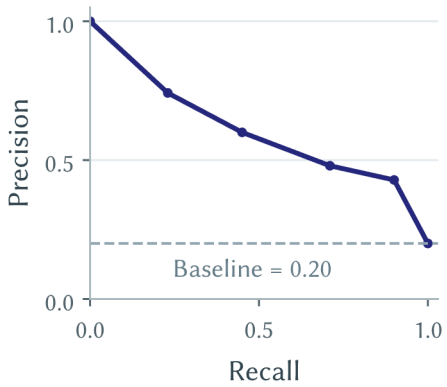
ROC vs. PR Curve: Two Evaluation Perspectives, Different Strengths

Both curves sweep every threshold. The choice between them depends on the dataset and the business question.

Illustrative ROC



Illustrative PR



ROC Curve + AUC

Axes: FPR (x) vs. TPR (y)

Summarises ranking across thresholds. The diagonal is the random-ranking baseline.

Best when:

Comparing ranking quality. For deployment, also consider class balance and error costs.

Caution:

A small FPR can still mean many false positives and low precision when negatives are numerous.

PR Curve

Axes: Recall (x) vs. Precision (y)

Shows the precision-recall tradeoff. The random-ranking baseline is the positive-class prevalence.

Best when:

Positives are rare or costly to miss, and false alarms also matter.

For the Telco dataset:

About 26.5% of customers churn. PR complements ROC by showing the reliability of churn flags.

Complete Evaluation Report: The Illustrative 2,000-Customer Example

Illustrative Evaluation Report

Metrics at θ = 0.50

Accuracy	**0.83**
Precision	**0.60**
Recall	**0.45**
F1 score	**0.51**
ROC AUC	**0.84**

Example cohort

2,000 customers, including 400 churners (20%)

Confusion matrix at θ = 0.50

180 TP 120 FP 220 FN 1,480 TN

Majority baseline

Accuracy 0.80 Recall 0.00 F1 0.00

AUC uses all scores

The other metrics depend on the threshold.

These are synthetic scores for teaching.

Tune the threshold on validation data.
Keep the test set for one final evaluation.

The judgment call:

No metric tells us whether this model is good enough to deploy. That requires comparing the cost of missed churners against the cost of the retention campaign, estimating the model's expected revenue impact, and deciding whether the business case is positive. Each of these steps requires domain knowledge the model cannot supply.

[Opening Question] We Have a Complete Evaluation — Why Would We Ever Use a Different Model?

Once a real logistic regression is fitted and evaluated, what could motivate replacing it with another model?

What if the pattern isn't linear?

The decision boundary

With the original features, logistic regression has a linear decision boundary. What if the data requires a curved or irregular boundary?

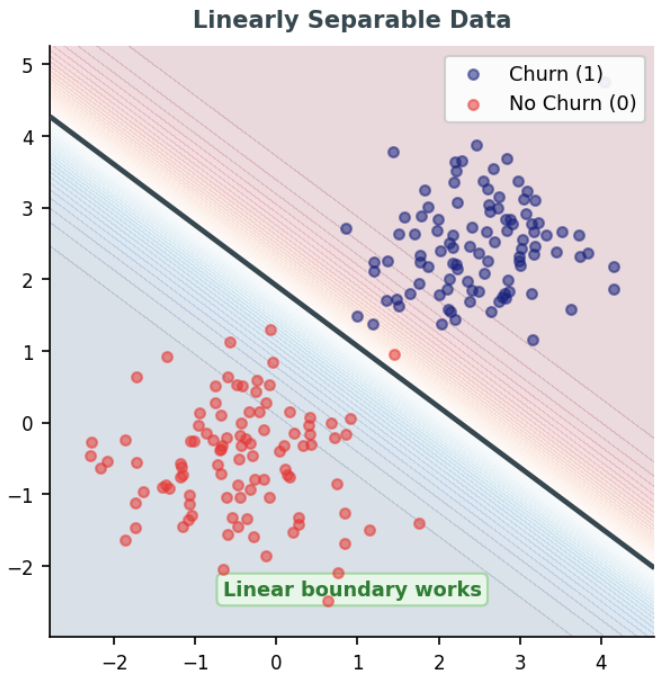
Interpretability vs. performance

A more complex model might score churners much better. Is the coefficient table worth the performance cost?

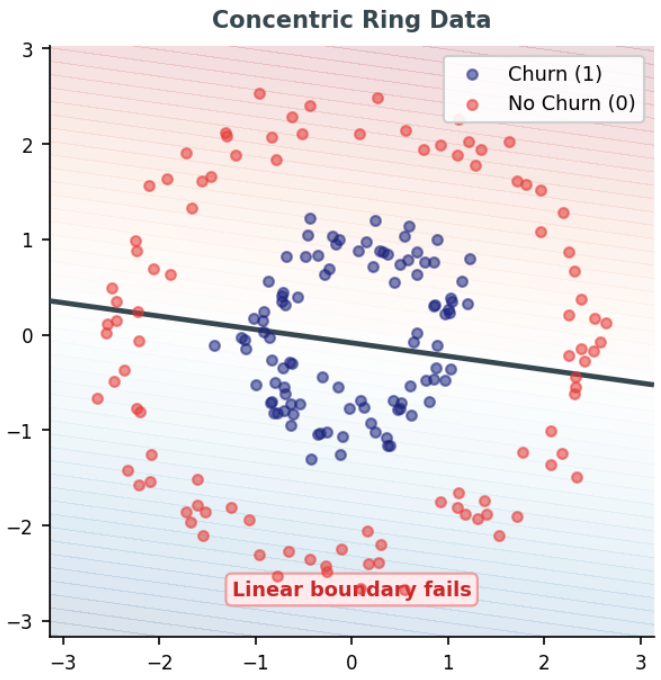
When does the model fail?

Even AUC = 0.84 leaves room to improve ranking. Inspect which positive–negative pairs receive incorrect or tied scores.

Linear Boundaries Have Limits — Trees Build Flexible, Piecewise Boundaries



A single hyperplane separates the classes cleanly.



No straight line can separate inner from outer ring.

Logistic regression boundary

With these two original features, the model has a straight boundary. Nonlinear feature engineering can produce a curved boundary in the original input space.

Decision tree boundary

Axis-aligned splits, such as tenure < 24 months and monthly charges > $70, form rectangular regions. More splits allow finer piecewise approximations.

The core tradeoff

A decision tree can fit non-linear patterns logistic regression misses. But it can also overfit — memorising the training data rather than generalising. More flexibility means more risk of overfitting.

CTA Demo 6: Fitting a Decision Tree and Comparing It to Logistic Regression

🤖 CTA Demo 6: Decision Tree — Fit, Visualise, Compare

Turn 1: Ask the CTA to fit `sklearn`'s `DecisionTreeClassifier(max_depth=4)` on the Telco training set and compare it on validation data. Also ask it to generate a visualisation of the first three levels of the tree using `sklearn`'s `plot_tree()`. We run both code blocks in Colab.

Turn 2 — Colab output: Display the metric comparison table from the Colab output: logistic regression vs. decision tree across accuracy, precision, recall, F1, AUC. Ask the CTA to summarise which model wins on each metric and on what grounds.

Turn 3: Ask the CTA whether the decision tree's coefficient equivalents — the split conditions — can be interpreted as easily as logistic regression's odds ratios. It should note the interpretability tradeoff explicitly.

Takeaway: The decision tree may outperform logistic regression on some metrics and underperform on others. Neither model family is universally best — compare validated performance and practical requirements in the specific business context.

Decision Tree vs. Logistic Regression: Compare Metrics and Practical Tradeoffs

Illustrative two-feature comparison at $\theta = 0.30$ (same example as the next slide)

Metric	Log. Reg.	Dec. Tree	Verdict
Threshold	0.30	0.30	Same cut-off
Precision	**0.38**	0.29	LR has higher precision
Recall	0.19	0.19	Both catch 19% of positives
F1 Score	**0.26**	0.23	LR slightly higher
AUC	**0.69**	0.65	LR ranks better overall

Logistic regression advantage:

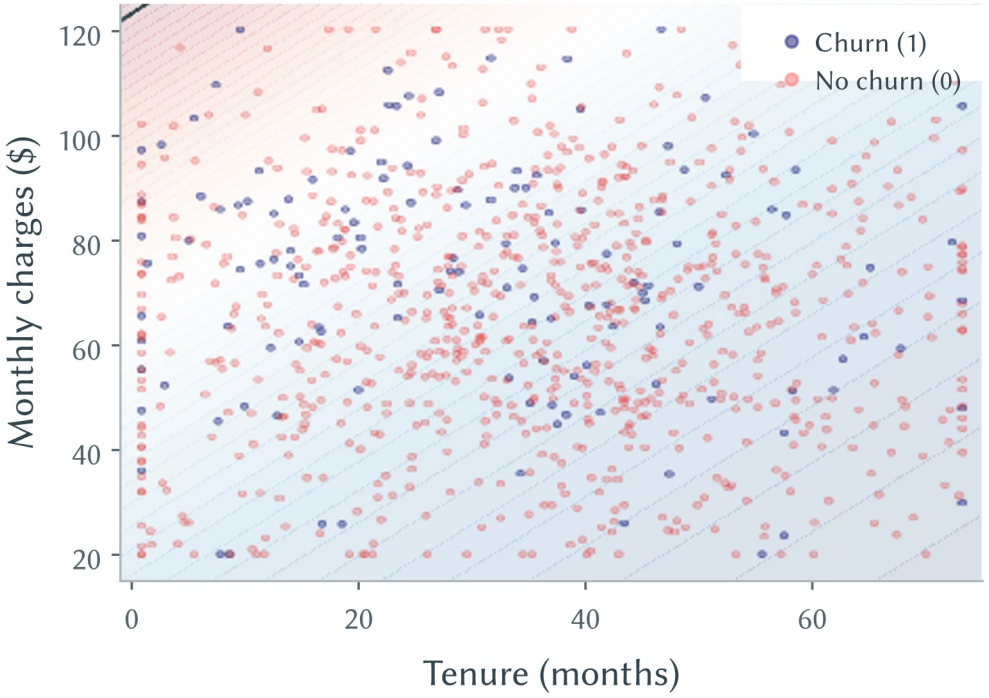
Coefficients express changes in modelled log-odds; exponentiating gives odds ratios for specified units and reference categories.

Decision tree advantage:

Split conditions produce human-readable rules. A depth-4 binary tree can have up to 16 leaf rules. Larger trees become harder to communicate and can be sensitive to small data changes.

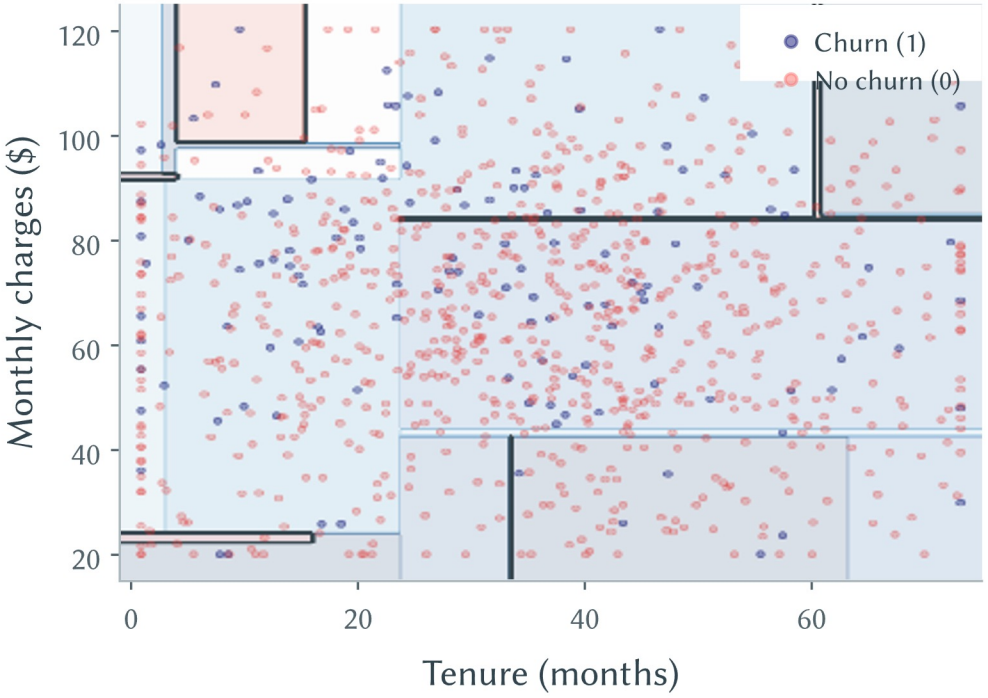
Logistic regression

Linear boundary



Decision tree

Piecewise rectangular boundaries



Metric / property	Logistic regression	Decision tree
ROC AUC	0.69	0.65
F1 (threshold = 0.30)	0.26	0.23
Precision	0.38	0.29
Recall	0.19	0.19
Interpretability	Odds ratios; specify units	If–then rules
Feature scaling	Recommended for penalised LR	Not required
Overfitting risk	Typically lower flexibility	Higher without depth limits

Separate illustrative two-feature comparison; metrics are retained rounded values.

Model Selection Is a Judgment Call — the Metrics Inform It, They Cannot Make It

Key Insight

Evaluation metrics quantify model behaviour on historical data. They do not quantify the value of deploying the model in a specific business context.

Choosing between logistic regression and a decision tree for the churn campaign requires answering questions the metrics cannot answer: How important is it that a marketing manager can explain why a customer was flagged? How frequently will the model be retrained? What is the regulatory environment around algorithmic decisions?

Each of these questions requires domain expertise, stakeholder knowledge, and institutional context — not a higher AUC.

Explainability

Can stakeholders understand and audit the model's decisions?

Maintenance

How much retraining and monitoring does deployment require?

Regulation

Are there legal constraints on which models can be used for these decisions?

Learning Objectives Revisited: What You Should Now Be Able to Do

LO1 ✓
Explain the range limitation of linear regression on binary targets and what the sigmoid solves

LO2 ✓
Fit logistic regression and interpret coefficients as odds ratios using the log-odds translation chain

LO3 ✓
Evaluate classifiers using precision, recall, and F1 — and explain why accuracy misleads for imbalanced classes

LO4 ✓
Create and interpret ROC and precision-recall curves, including the probabilistic meaning of AUC

LO5 ✓
Select a classification threshold by specifying an explicit cost function for false positives and false negatives

LO6 ✓
Collaborate with the CTA to fit, evaluate, and compare classifiers — while identifying what each output requires human judgment to interpret

LO7 ✓
Articulate the boundary between what evaluation metrics can determine and what requires domain expertise

Technically Intriguing Questions: What This Lecture Does Not Cover

Each of these is a natural extension — explored further in later lectures or independent study.

Maximum likelihood and model fitting → *try asking the CTA after class*
How does `sklearn` fit logistic regression? It numerically minimises penalised negative log-likelihood; the default solver is L-BFGS.

Regularisation (L1 and L2) → *try asking the CTA after class*
`sklearn`'s `LogisticRegression` applies L2 regularisation by default (`C=1.0`). What does this do to the coefficients, and how should C be chosen?

Multiclass classification → *try asking the CTA after class*
Logistic regression generalises to K classes via the softmax function. Precision, recall, and F1 extend via micro/macro averaging — which choice matters and why.

Project Proposal, Homework 2, and Mid-term

See Moodle for the confirmed schedule.

Check the current course announcements for submission deadlines, release dates, and exam details.

Questions?

CONTACT INFORMATION



yangliu.mit.edu

Instructor: Prof. Yang Liu

Email: yliu12@hku.hk

Office Hours: 5:00 - 5:50 PM, Thursdays

Location: CB-714, Chow Yei Ching Building

Let me know if you want remote!

🤖 **CTA is your 24/7 interactive and responsive learning partner!**

TA contacts

Tuesday 4-5PM

Yushi Du:
u3012725@connect.hku.hk
https://us04web.zoom.us/j/78561597481

Zhengyang Hu:
u3010250@connect.hku.hk
https://hku.zoom.us/j/9021481973

Thursday 3-4PM

Jun Dai:
jundai332@connect.hku.hk
https://rochester.zoom.us/j/97202452759

Lulu Cai:
u303011209@connect.hku.hk
https://hku.zoom.us/j/5010926726

TA office hours on Zoom! Two slots (1 hour each) on Tuesdays and Thursdays