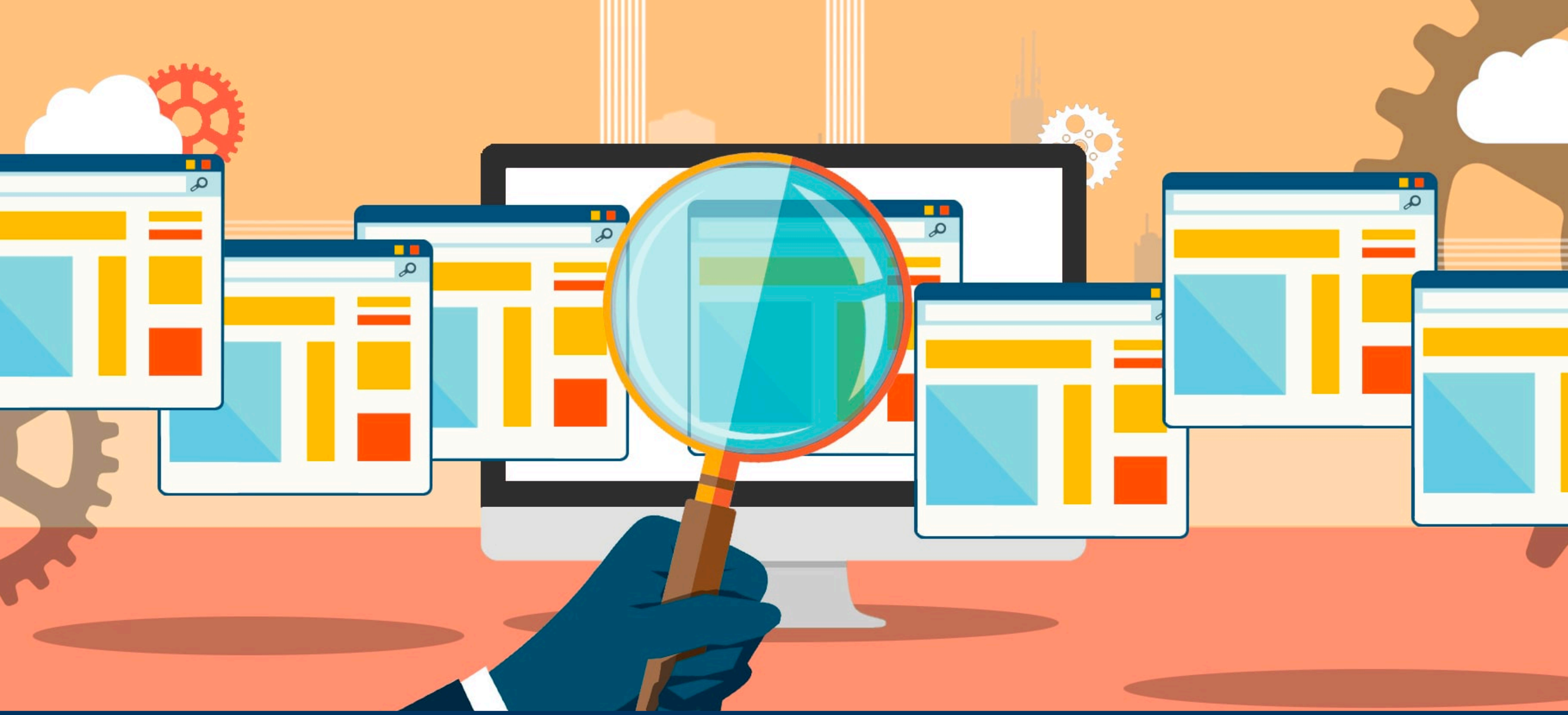
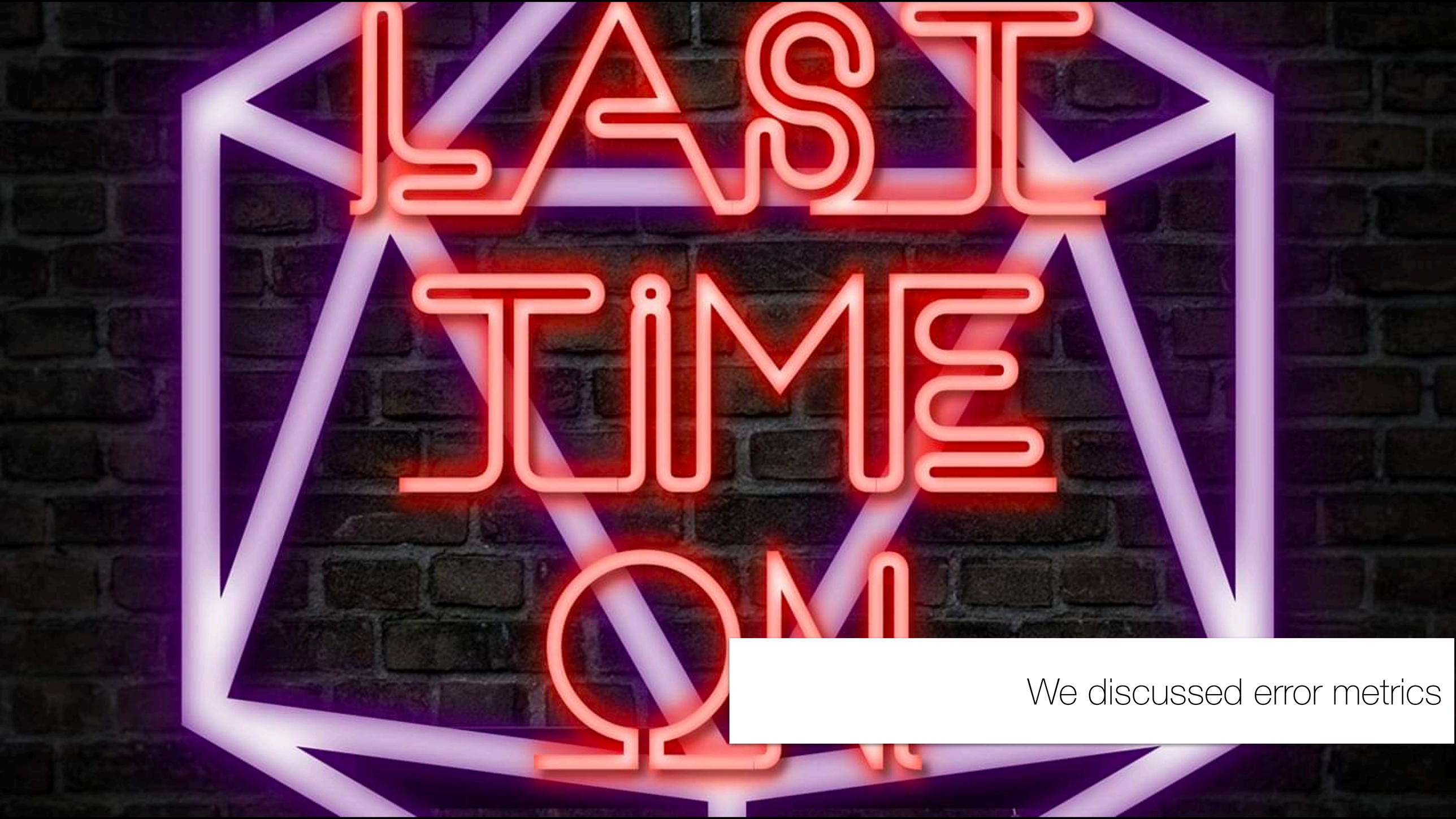


Evaluation in Supervised Learning

INST414 - Data Science Techniques



We discussed error metrics

How do you **evaluate** the **performance/quality** of your model?

Depends on classification or regression

Many, many evaluation metrics exist

Model Evaluation – Classification

Most simplistic evaluation: Accuracy

How often did you predict the right label?

Accuracy = $\frac{1}{|X|} \sum_i f(x_i) == y_i$

|X| = # of samples

Was the prediction the same as the actual label?

Model Evaluation – Regression

Common metric: average square of delta between true and predicted value

"Error" = predicted value - true value

Mean Squared Error =

$$\frac{1}{|X|}\sum_{i}(f(x_i)-y_i)^2$$

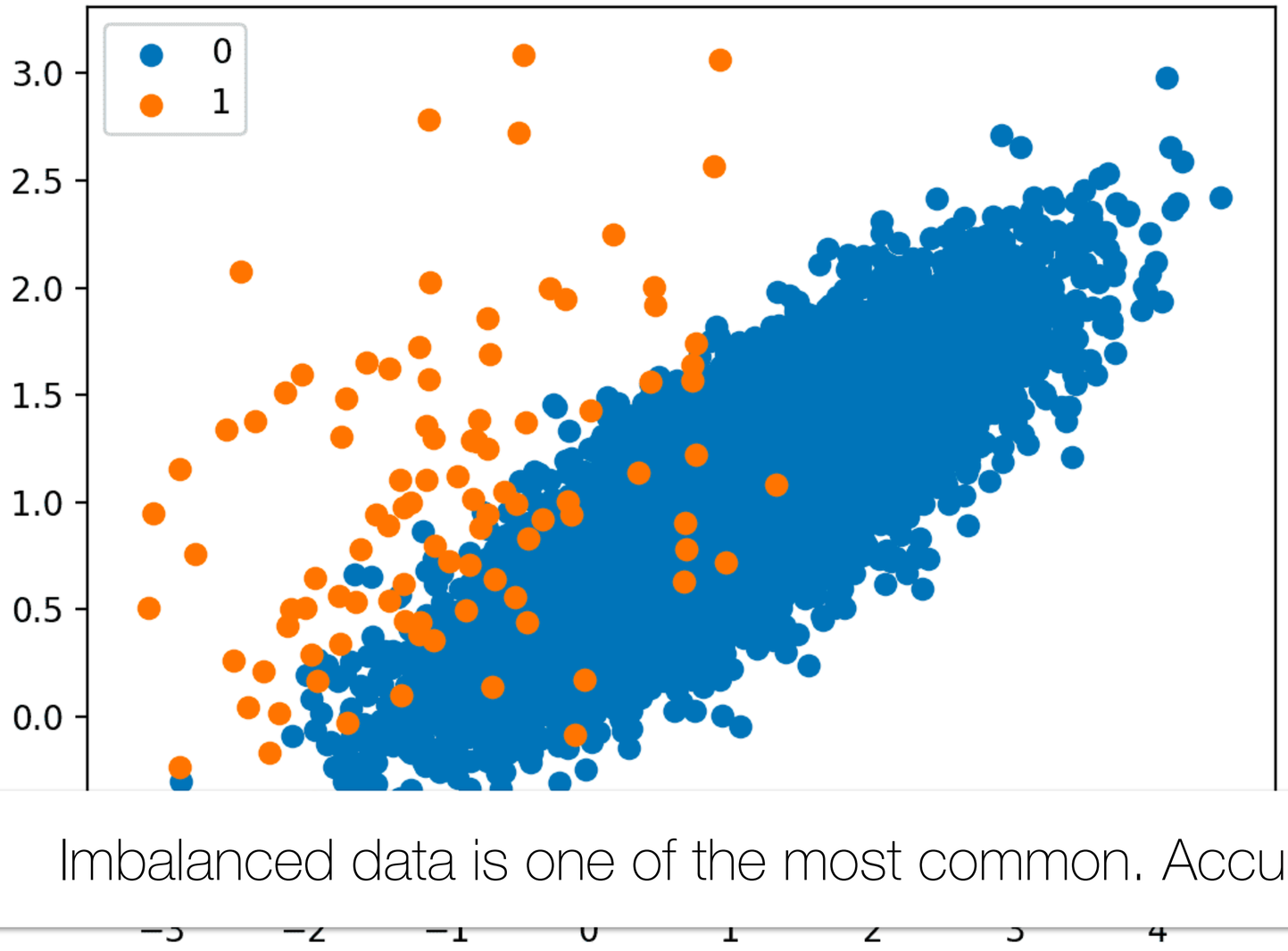
|X| = # of samples

Delta between true and predicted value

do you **evaluate** the **performance/quality** of your model?

Depends on classification or regression

Many, many evaluation metrics exist



Imbalanced data is one of the most common. Accuracy means less here

This Module's Learning Objectives

Evaluation in Supervised Learning

Describe why accuracy may be a poor metric for imbalanced data

Calculate precision and recall for a supervised learning model

Differentiate macro- and micro-averaging for multi-class evaluation

Use k-fold cross-validation to estimate average performance

This Module's Learning Objectives

Evaluation in Supervised Learning

Describe why accuracy may be a poor metric for imbalanced data

Calculate precision and recall for a supervised learning model

Differentiate macro- and micro-averaging for multi-class evaluation

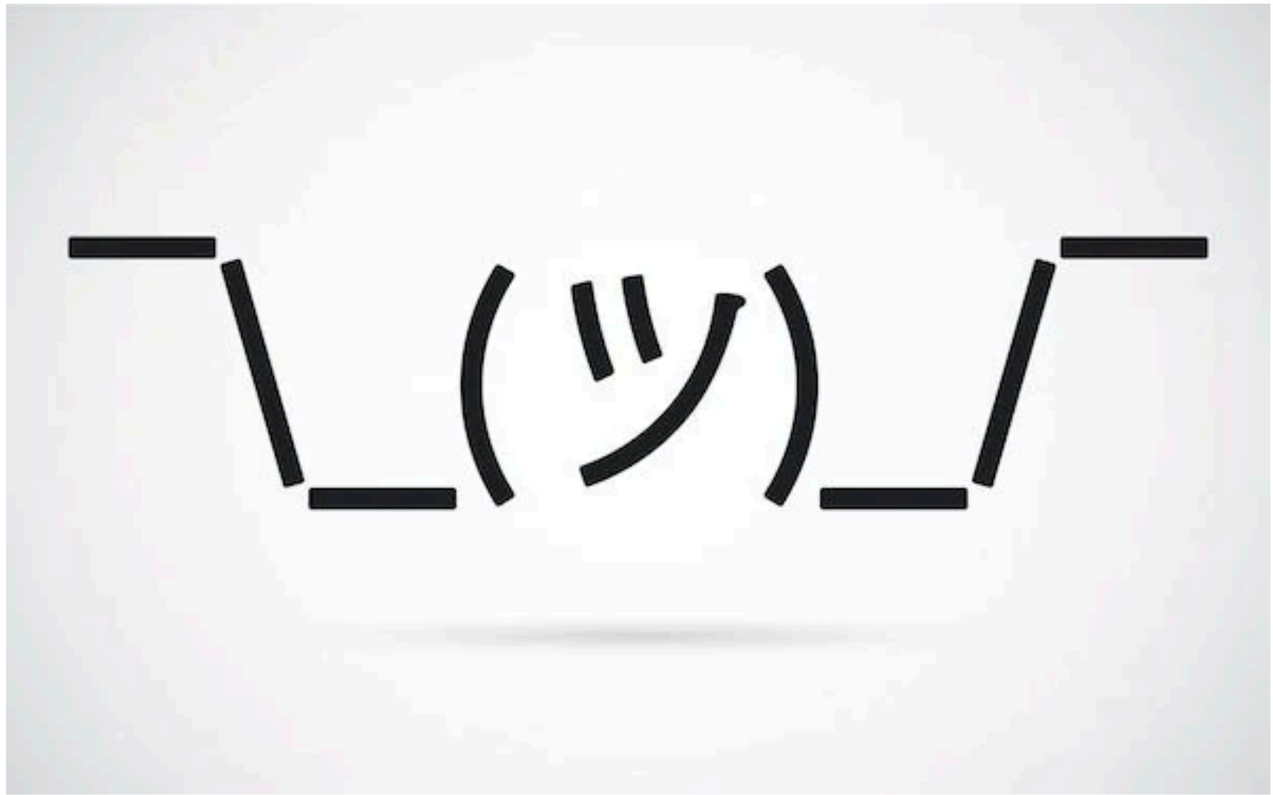
Use k-fold cross-validation to estimate average performance

Why accuracy score isn’t always good.

- Tumor type classification based on CT image
- You have 1,000 CT images with potential for tumors
  - Benign (“Negative”): 998 out of 1,000
  - Malicious (“Positive”): 2 out of 1,000
- Accuracy = # correct prediction / # total instances
- Your classifier achieves 0.98 accuracy
  - Seems good, right? 👍🏻
  - But is it, really? 🤔

Why accuracy score isn't always good.

- A single accuracy metric is not super informative
  - Need comparisons for context!
  - But compare against what?



Why accuracy score isn't always good.

- Could compare to other models (kNN, logistic regression, etc.)
- Compare with a baseline / "dummy classifier"
- Simple Baseline: Zero Rule
  - Always predict the most common label

$$\text{Accuracy} = \frac{Correct}{Total}$$

$$Accuracy(\text{Hypothetical Classifier}) = 0.98$$

$$Accuracy(\text{Zero Rule, All-Negative}) = \frac{998}{1000}$$

Accuracy(Zero-Rule) > Accuracy(Classifier), 👎🏻 👎🏻 👎🏻

Dummy Classifiers as Sanity Check / Baseline

- Dummy classifiers ignore input data and makes classifications
- Common baselines:
  - Majority: predicts the most frequent label in the training set.
  - Random: generates predictions at random
  - Others?

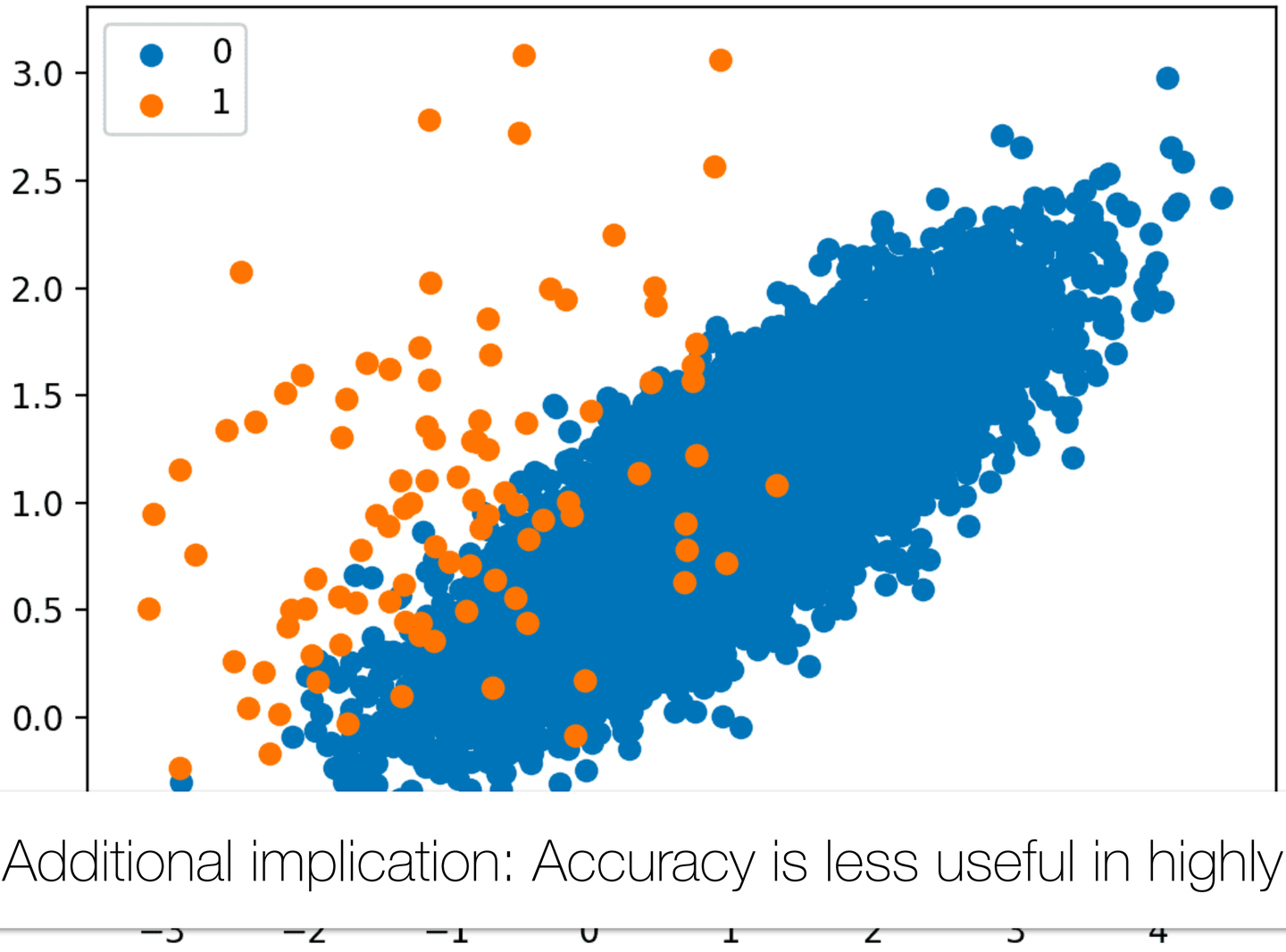
Dummy Classifiers as Sanity Check / Baseline

- If a classifier accuracy is close to the baseline, might be:
  - Uninformative, erroneous, or missing features
  - Poor choice of learning model
  - Large class imbalance

Does accuracy of zero-rule > tumor detection classifier matter?

We will never identify a tumor with zero-rule

Then what's the point of the classifier?



Additional implication: Accuracy is less useful in highly imbalanced datasets

Are there metrics that are more robust to imbalance?

This Module's Learning Objectives

Evaluation in Supervised Learning

Describe why accuracy may be a poor metric for imbalanced data

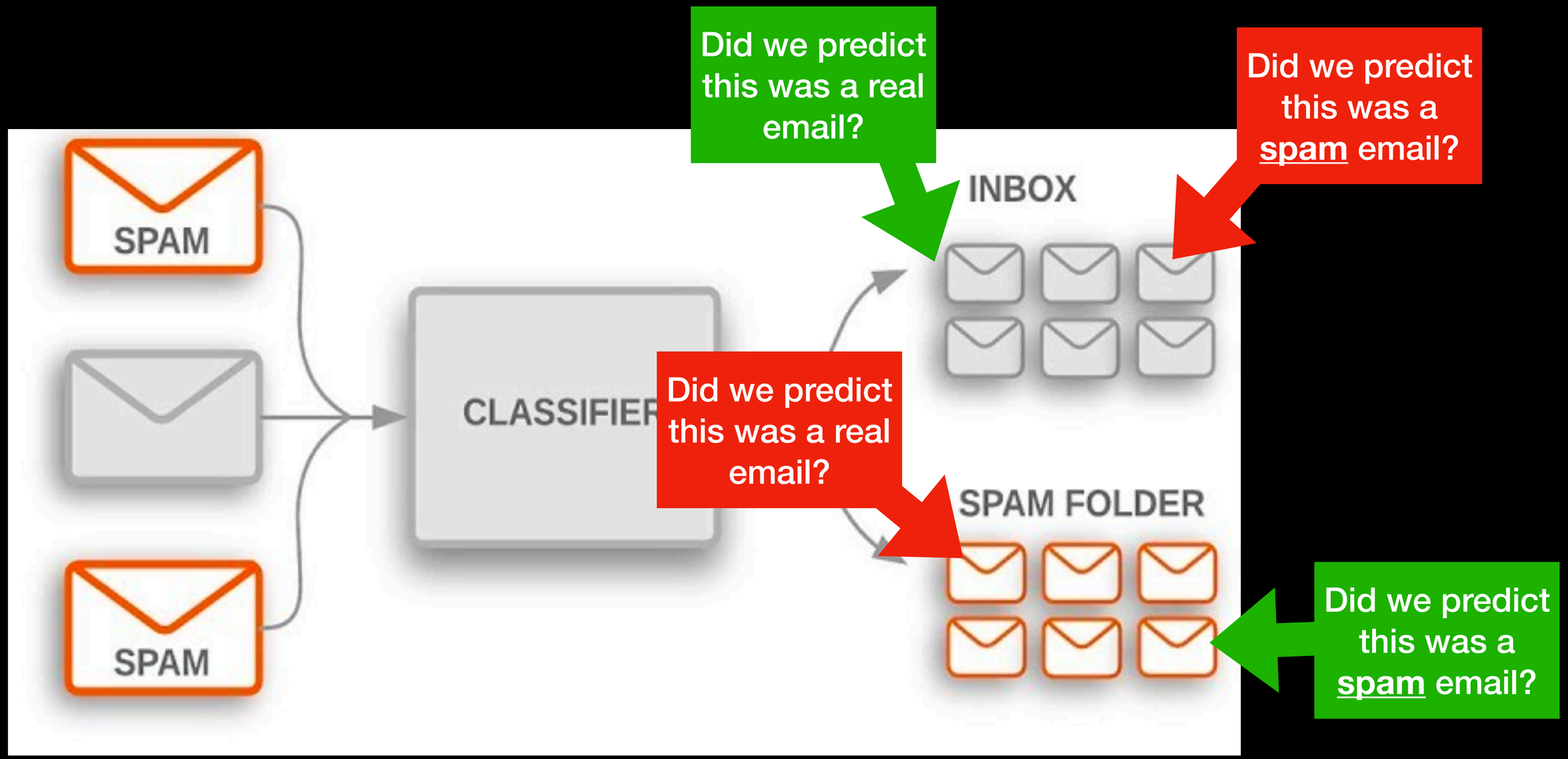
Calculate precision and recall for a supervised learning model

Differentiate macro- and micro-averaging for multi-class evaluation

Use k-fold cross-validation to estimate average performance

Diving into predictions. What might happen?

Four possibilities exist in our predictions…



We Have Seen Extremely Imbalanced Data

- Dive into the actual predictions:
  - “True Positive” = A sample is actually a positive instance of our class
  - “False Positive” = A sample is classified as positive but is actually negative
  - “True Negative” = A sample is actually a negative instance of our class
  - “False Negative” = A sample is classified as negative but is actually positive

Confusion Matrix in Classification

	Predicted Positive	Predicted Negative
Actually Positive		
Actually Negative		

Confusion Matrix in Classification

	Predicted Positive	Predicted Negative
Actually Positive	True Positive (TP)	
Actually Negative		

Confusion Matrix in Classification

	Predicted Positive	Predicted Negative
Actually Positive	True Positive (TP)	
Actually Negative		True Negative (TN)

Confusion Matrix in Classification

	Predicted Positive	Predicted Negative
Actually Positive	True Positive (TP)	
Actually Negative	False Positive (FP)	True Negative (TN)

Confusion Matrix in Classification

	Predicted Positive	Predicted Negative
Actually Positive	True Positive (TP)	False Negative (FN)
Actually Negative	False Positive (FP)	True Negative (TN)

Confusion Matrix in Classification

	Predicted Positive	Predicted Negative
Actually Positive	True Positive (TP)	False Negative (FN)
Actually Negative	False Positive (FP)	True Negative (TN)

Type I error

Type II error

Confusion Matrix in Classification

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN}$$

	Predicted Positive	Predicted Negative
Actually Positive	True Positive (TP)	False Negative (FN)
Actually Negative	False Positive (FP)	True Negative (TN)

$$\text{Precision} = \frac{TP}{TP + FP}$$

Of the things we said were positive, how correct were we?

$$\text{Recall} = \frac{TP}{TP + FN}$$

Of ALL the things that ARE positive, how many did we find?

Revisiting Tumor Classification

- 1,000 CT images with potential for tumors
  - Benign (“Negative”): 998 out of 1,000
  - Malicious (“Positive”): 2 out of 1,000

$$Accuracy(\text{Hypothetical Classifier}) = 0.98$$

$$Accuracy(\text{Zero Rule, All-Negative}) = \frac{998}{1000}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

Zero-Rule Baseline

	Predicted Positive	Predicted Negative
Actually Positive	0	2
Actually Negative	0	998

$$\text{Recall} = \frac{0}{0+2} = 0 \qquad \text{Precision} = \frac{0}{0+0} = \text{undef}$$

Hypothetical Classifier

	Predicted Positive	Predicted Negative
Actually Positive	2	0
Actually Negative	20	978

$$\text{Recall} = \frac{2}{2+0} = 1 \qquad \text{Precision} = \frac{2}{2+20} = 0.091$$

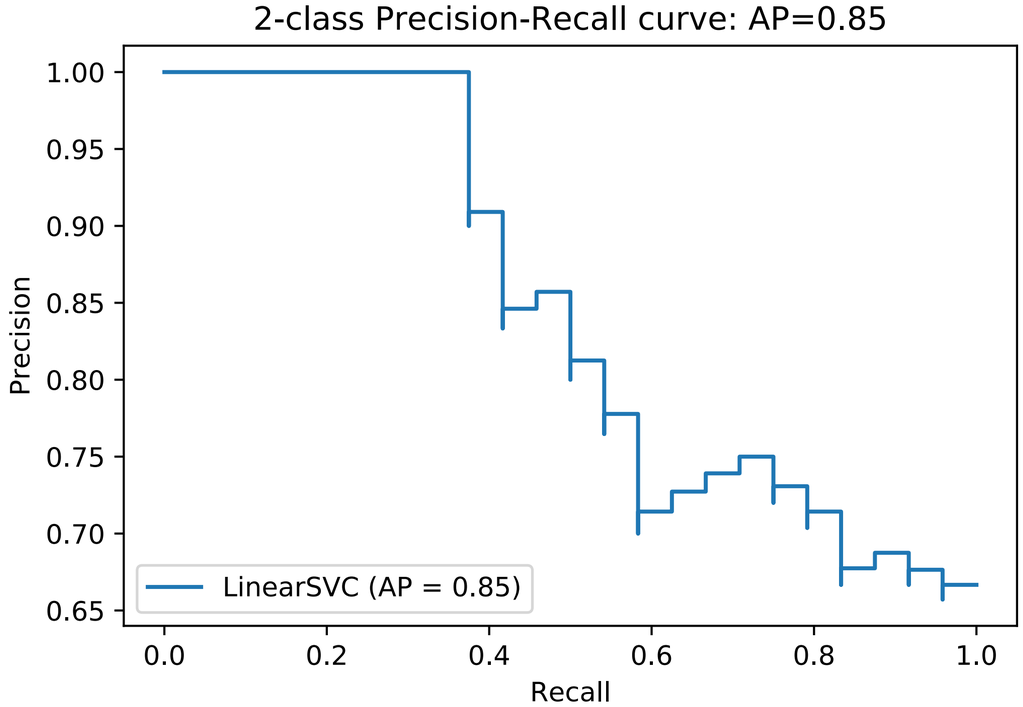
$$\text{Recall} = \frac{TP}{TP + FN}$$

What does *recall = 1.0* mean?

$$\text{Precision} = \frac{TP}{TP + FP}$$

What does *precision = 1.0* mean?

Precision-Recall Tradeoff



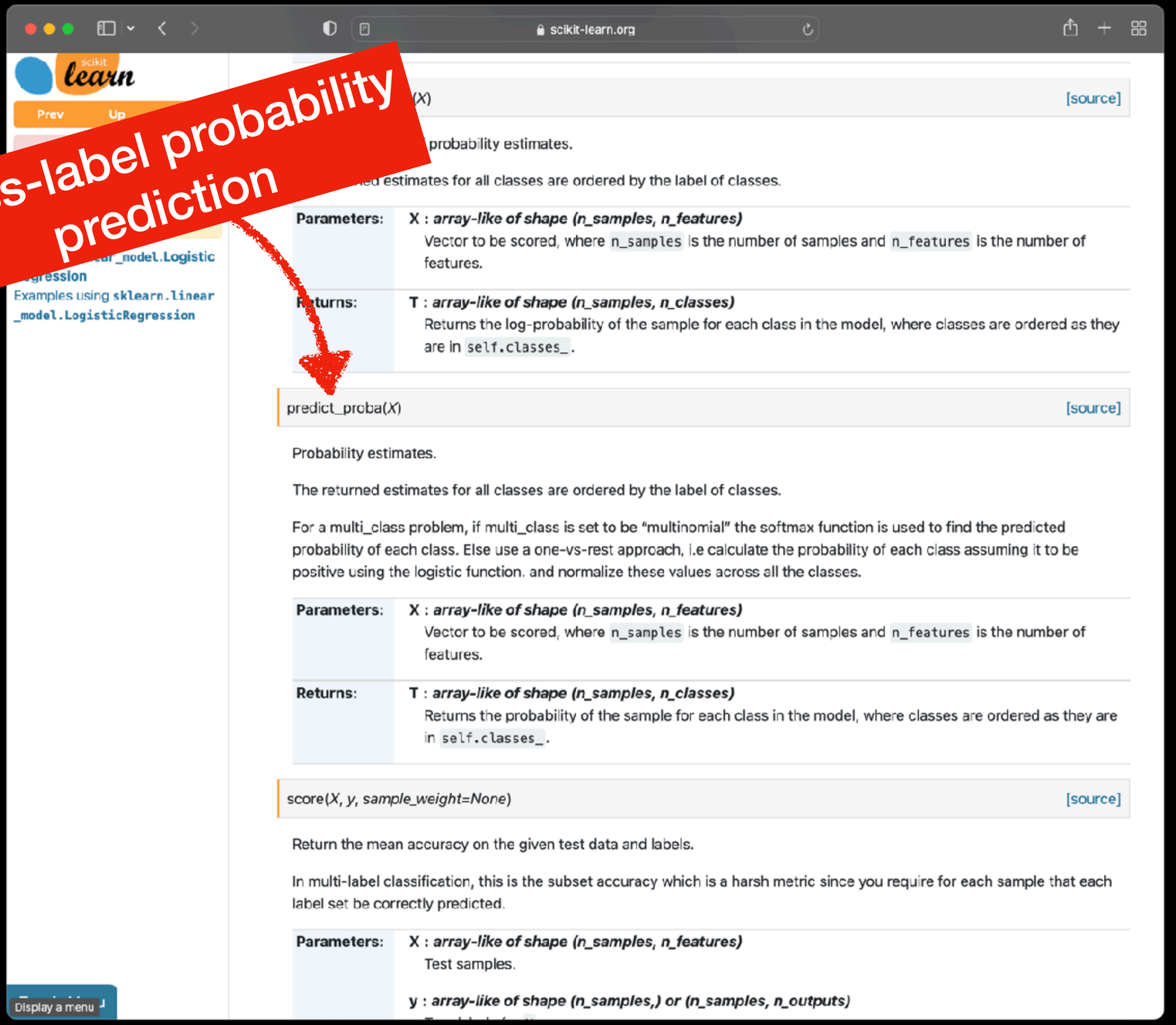
- Recall-oriented tasks:
  - Tumor detection
  - Legal discovery
  - (False Negatives are more costly.)
- Precision-oriented tasks:
  - Search engine ranking
  - Sentiment classification
  - Customer-facing tasks (users remember failures!)
  - (False Positives are more costly.)

Combining Precision and Recall with F-score

- $F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$
- $F_\beta = \frac{(1 + \beta^2) \times \text{Precision} \times \text{Recall}}{\beta^2 \times \text{Precision} + \text{Recall}}$
- β allows adjustment of the metric to control the emphasis on recall vs precision:
  - Precision-oriented: small β
  - Recall-oriented: large β

Ranking in Classification Prediction

- Instead of predicting class labels (0/1), we predict the probability of belonging to positive class.


scikit-learn.org
Class-label probability prediction
Prev
Up
Examples using sklearn.linear_model.LogisticRegression
(X) [source]
probability estimates.
estimates for all classes are ordered by the label of classes.
Parameters: X : array-like of shape (n_samples, n_features)
Vector to be scored, where n_samples is the number of samples and n_features is the number of features.
Returns: T : array-like of shape (n_samples, n_classes)
Returns the log-probability of the sample for each class in the model, where classes are ordered as they are in self.classes_.
predict_proba(X) [source]
Probability estimates.
The returned estimates for all classes are ordered by the label of classes.
For a multi_class problem, if multi_class is set to be "multinomial" the softmax function is used to find the predicted probability of each class. Else use a one-vs-rest approach, i.e calculate the probability of each class assuming it to be positive using the logistic function. and normalize these values across all the classes.
Parameters: X : array-like of shape (n_samples, n_features)
Vector to be scored, where n_samples is the number of samples and n_features is the number of features.
Returns: T : array-like of shape (n_samples, n_classes)
Returns the probability of the sample for each class in the model, where classes are ordered as they are in self.classes_.
score(X, y, sample_weight=None) [source]
Return the mean accuracy on the given test data and labels.
In multi-label classification, this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted.
Parameters: X : array-like of shape (n_samples, n_features)
Test samples.
y : array-like of shape (n_samples,) or (n_samples, n_outputs)
Display a menu

Ranking in Classification Prediction

- Instead of predicting class labels (0/1), we predict the probability of belonging to positive class.
- Many classification algorithm can predict such probability.
  - Logistic Regression (Sigmoid Function)
  - Random Forest (% of votes)
  - Implemented as `decision_function` or `predict_proba` method in scikit-learn.
- Changing the threshold affects the prediction:
  - Default: 50% for binary classification.
  - Higher threshold results in a more conservative classifier (less "willing" to predict positive).

Regression Metrics

- Typically R^2 score is enough
  - Reminder: evaluate how well future instances will be predicted
  - Between 0 (constant prediction) and 1 (best possible score).
- Alternatives:
  - Mean Absolute Error (MAE, absolute difference of target & predicted values)
  - Mean Squared Error (MSE, squared difference)
  - Median Absolute Error (robust to outliers)
- Also helpful to check baselines/“Dummy Regressors”
  - mean/median
  - Random

Dealing with Data Imbalance

- When classes are severely imbalanced, model fitting is difficult
- Re-sampling the data can make it balanced
  - Down-sample large classes
  - Up-sample small classes
- Consider using ranking instead of classification models

1. Introduction

1.1. API's of imbalanced-learn samplers

The available samplers follows the scikit-learn API using the base estimator and adding a sampling functionality through the `sample` method:

Estimator: The base object, implements a `fit` method to learn from data, either:

```
estimator = obj.fit(data, targets)
```

Resampler: To resample a data sets, each sampler implements:

```
data_resampled, targets_resampled = obj.fit_resample(data, targets)
```

Imbalanced-learn samplers accept the same inputs that in scikit-learn:

- **data**:
  - 2-D **list**,

Whole Python package for dealing with imbalanced data, called imblearn

- **targets**:
  - 1-D **numpy.ndarray**,
  - **pandas.Series**.

Edit this page

This Module's Learning Objectives

Evaluation in Supervised Learning

Describe why accuracy may be a poor metric for imbalanced data

Calculate precision and recall for a supervised learning model

Differentiate macro- and micro-averaging for multi-class evaluation

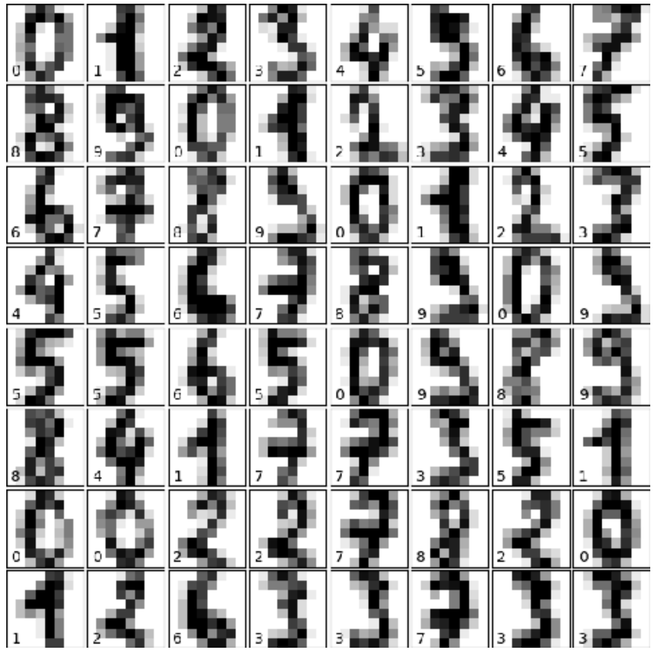
Use k-fold cross-validation to estimate average performance

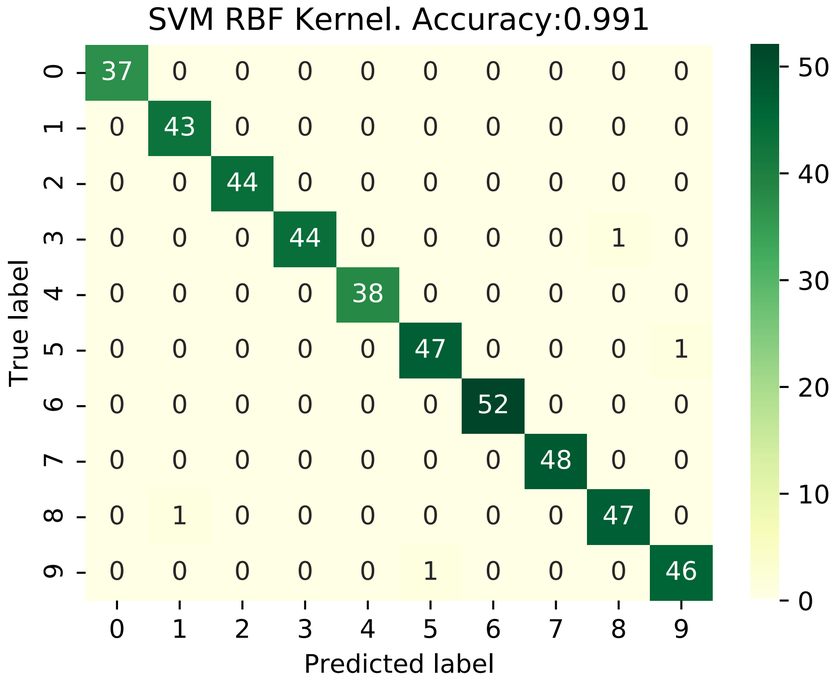
Multi-Class Classification

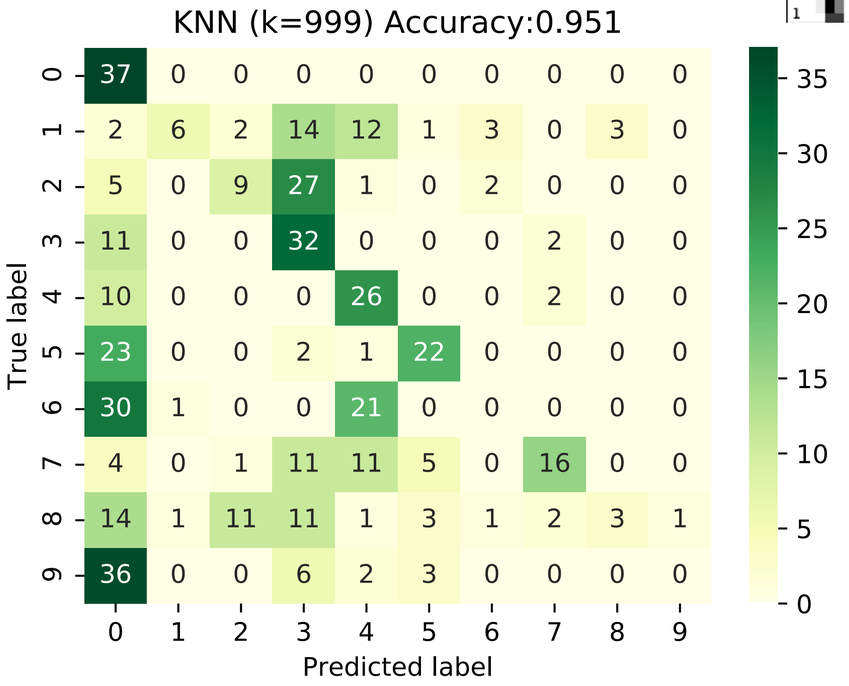
- An extension to binary classification.
- Can use multiple binary classifications (one vs. other)

- Multi-class evaluation is an extension of the binary case
  - A collection of true vs. predicted binary outcomes, one per class
  - Confusion metrics are especially useful

Multi-Class Confusion Matrix



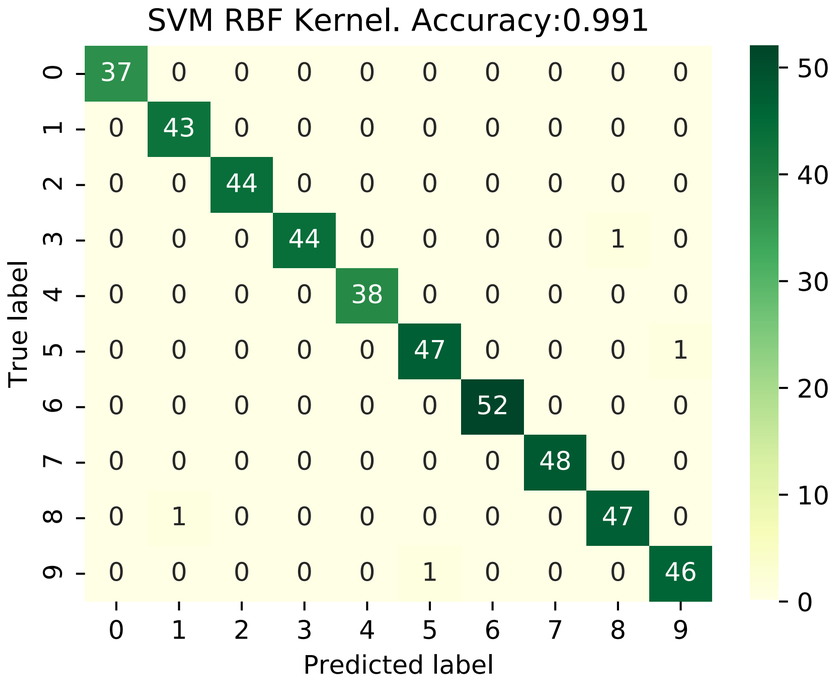




Multi-Class Classification

- Overall evaluation metrics are average across classes
  - But there are different ways to average

Average across this matrix



Micro-Average

- Each instance has equal weights.
  - Aggregate outcomes across all classes.
  - Compute metric with aggregated outcomes.

Class	Predicted Class	Correct?
Orange	Lemon	0
Orange	Lemon	0
Orange	Apple	0
Orange	Orange	1
Orange	Apple	0
Lemon	Lemon	1
Lemon	Apple	0
Apple	Apple	1
Apple	Apple	1

Micro Average Recall:
4 / 9 = 0.44

Largest classes have most influence!

Macro-Average

- Each class has equal weights.
  - First compute metric within each class.
  - Then average resulting metrics across classes.

Class	Predicted Class	Correct?
Orange	Lemon	0
Orange	Lemon	0
Orange	Apple	0
Orange	Orange	1
Orange	Apple	0
Lemon	Lemon	1
Lemon	Apple	0
Apple	Apple	1
Apple	Apple	1

Class	Recall
Orange	1/5 = 0.2
Apple	2/2 = 1.0
Lemon	1/2 = 0.5

Macro Average Recall:
(0.2 + 0.5 + 1.0) / 3 = 0.57

All classes have the same influence!

Imbalance and Macro- vs. Micro-Averaging

- If the classes have about the same number of instances, macro- and micro-average will be about the same.
- If some classes are much larger (more instances) than others, and you want to:
  - Weight your metric toward the largest ones, use micro-averaging.
  - Weight your metric toward the smallest ones, use macro-averaging.
- Rules of Thumb:
  - If micro-average << macro-average, then examine the larger classes.
  - If micro-average >> macro-average, then examine the smaller classes.

This Module's Learning Objectives

Evaluation in Supervised Learning

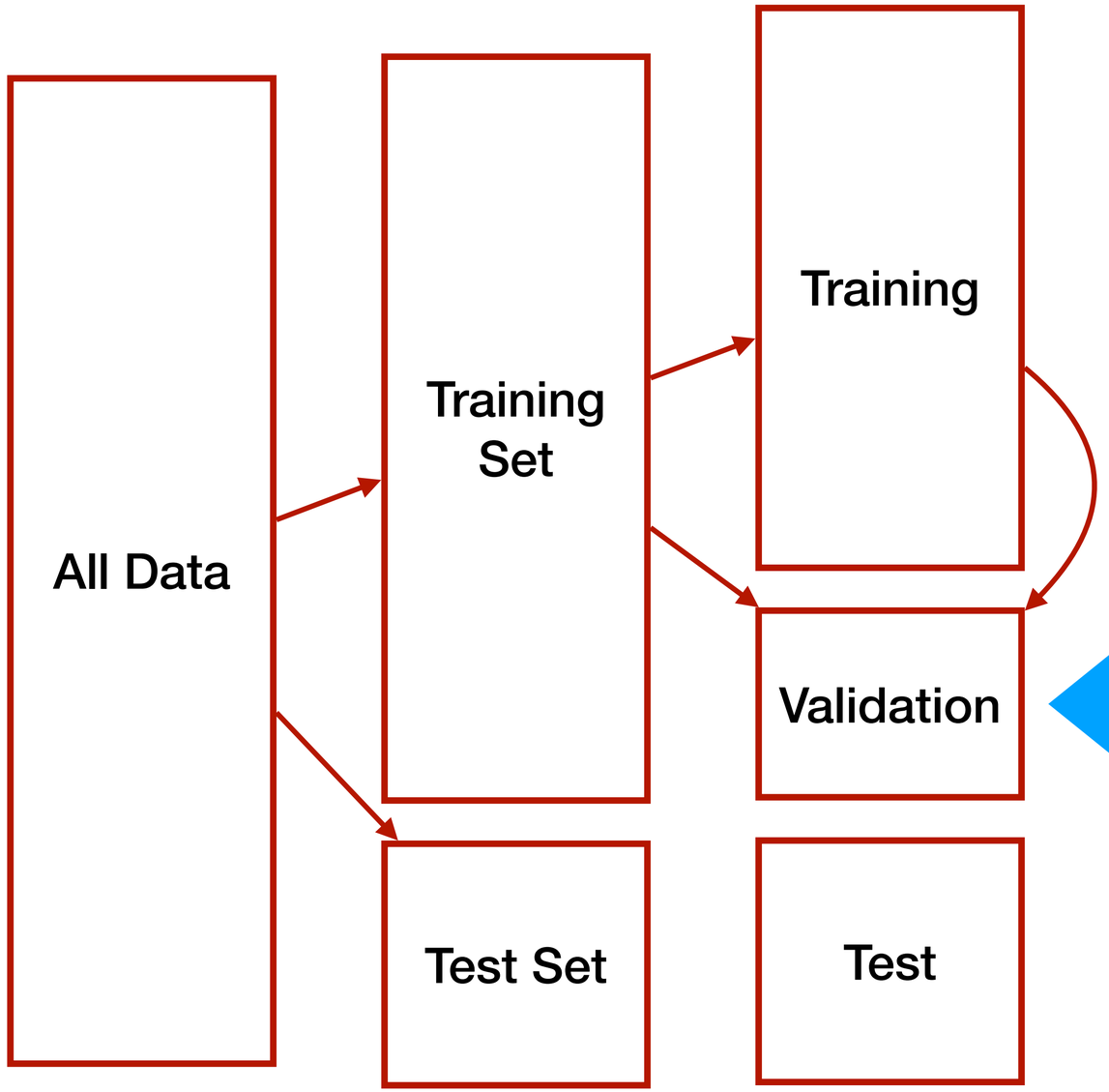
Describe why accuracy may be a poor metric for imbalanced data

Calculate precision and recall for a supervised learning model

Differentiate macro- and micro-averaging for multi-class evaluation

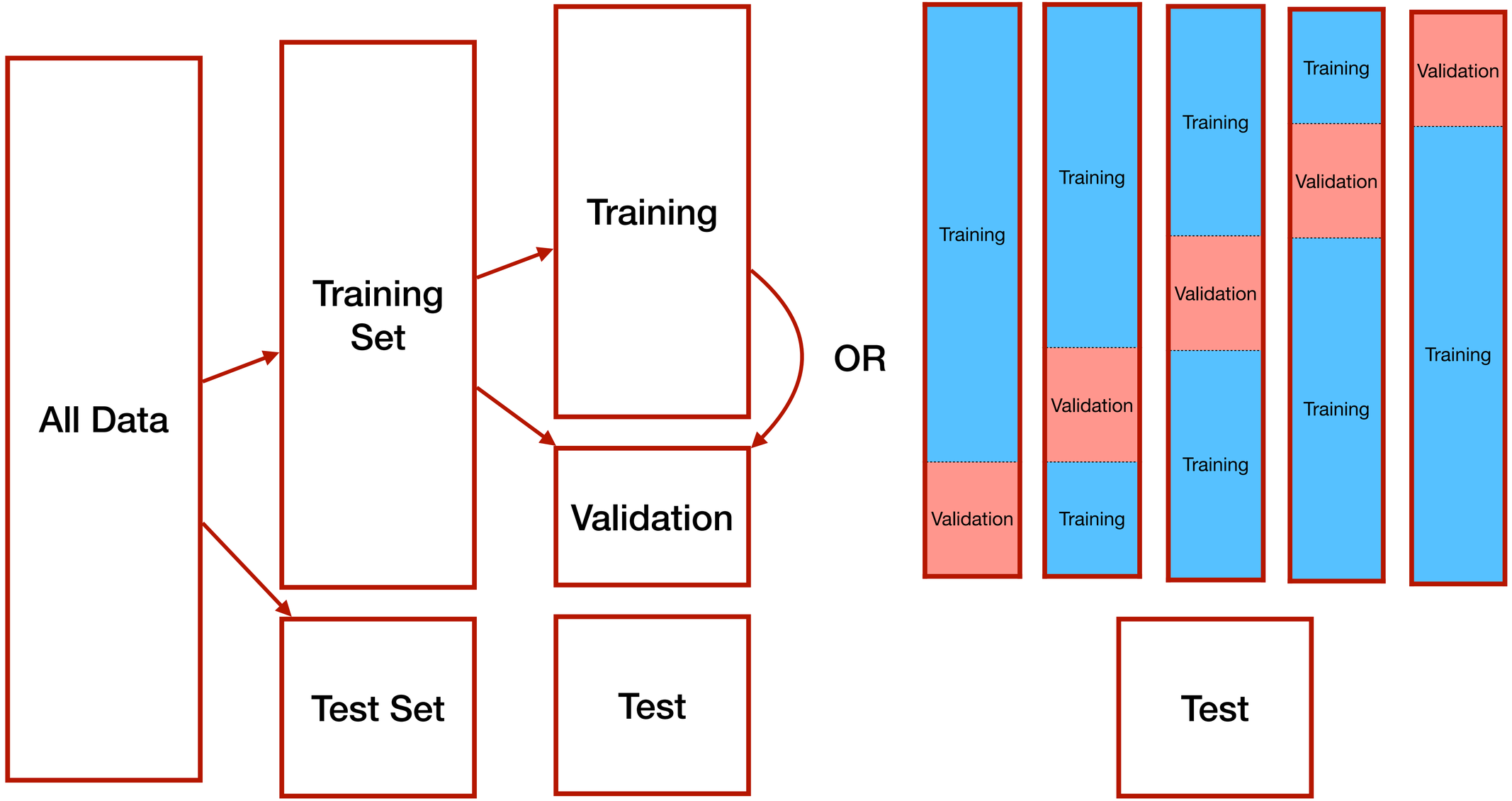
Use k-fold cross-validation to estimate average performance

Model Selection: Training-Validation-Test Split

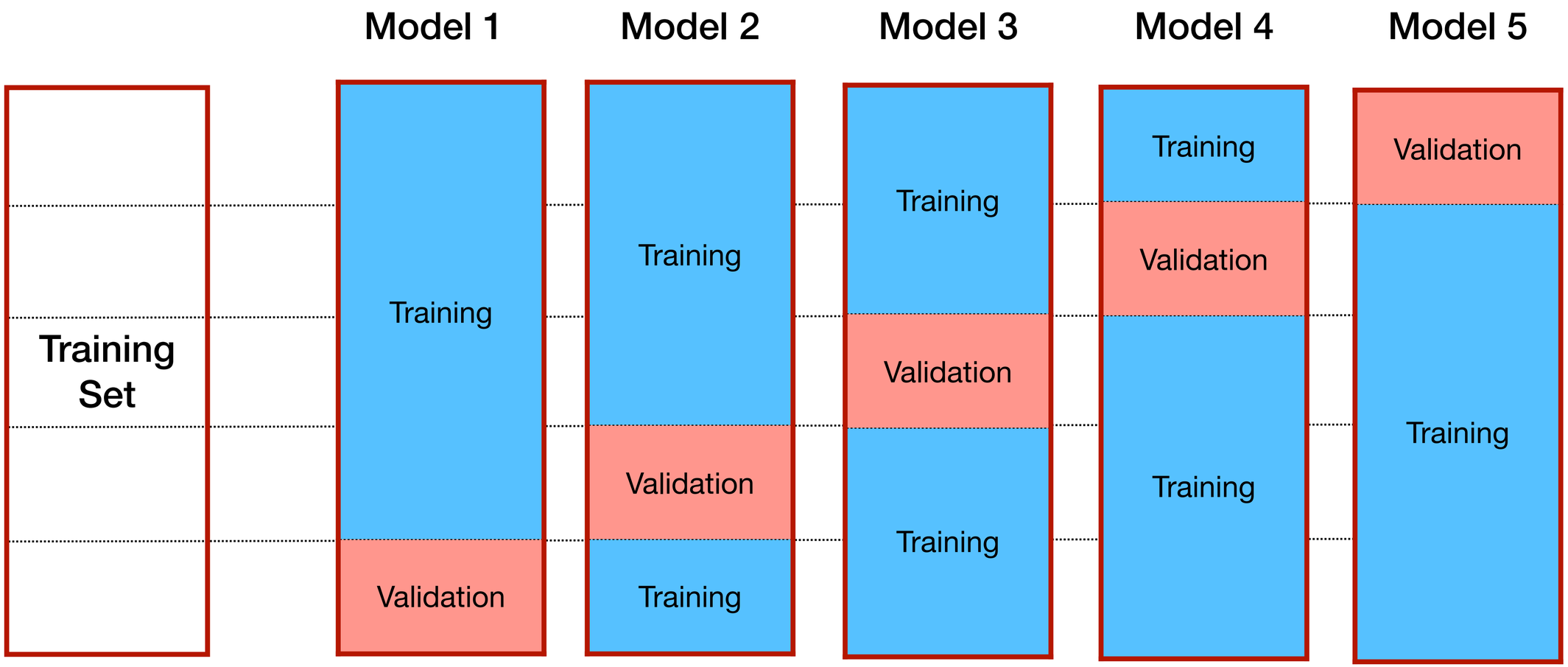


Now we lose this data from our training set… 👎

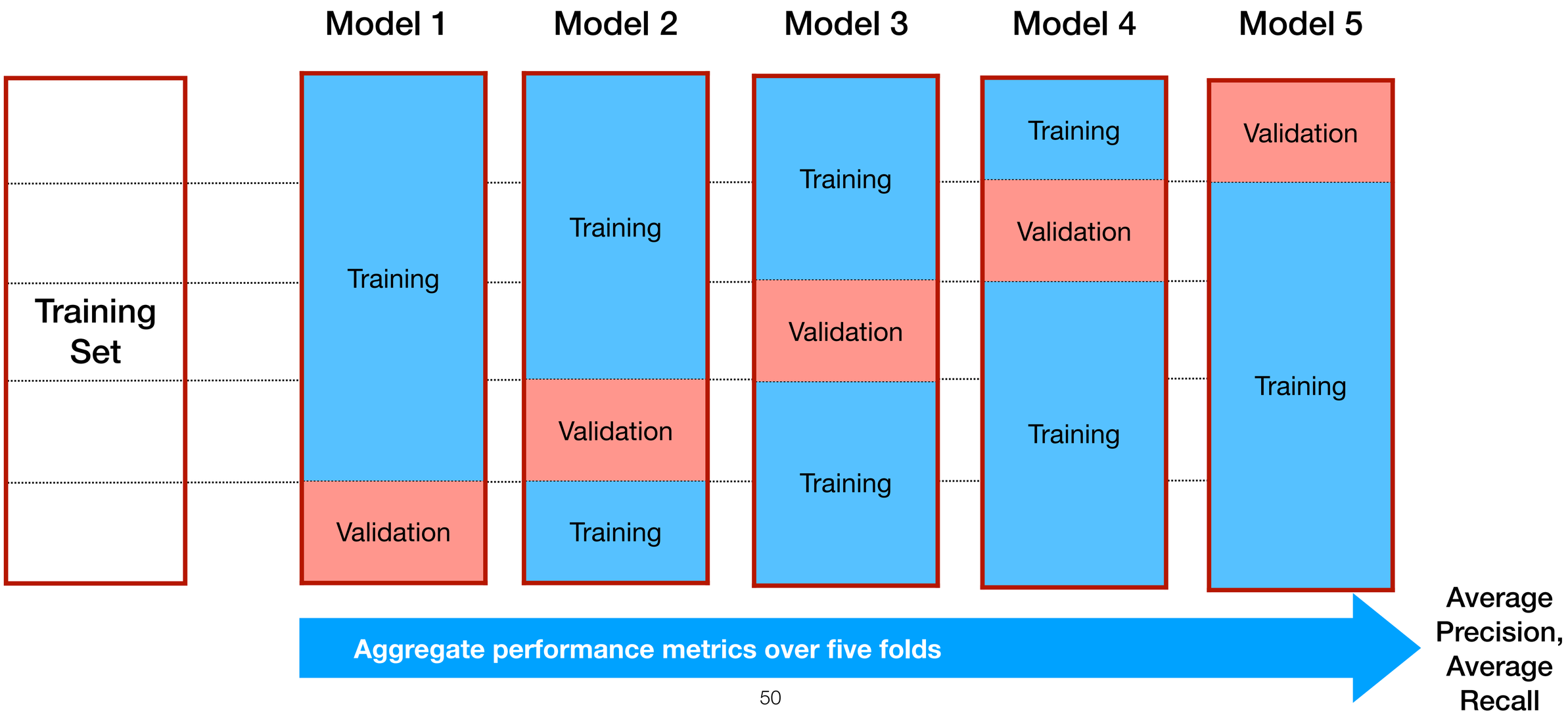
Model Selection: Training-Validation-Test Split

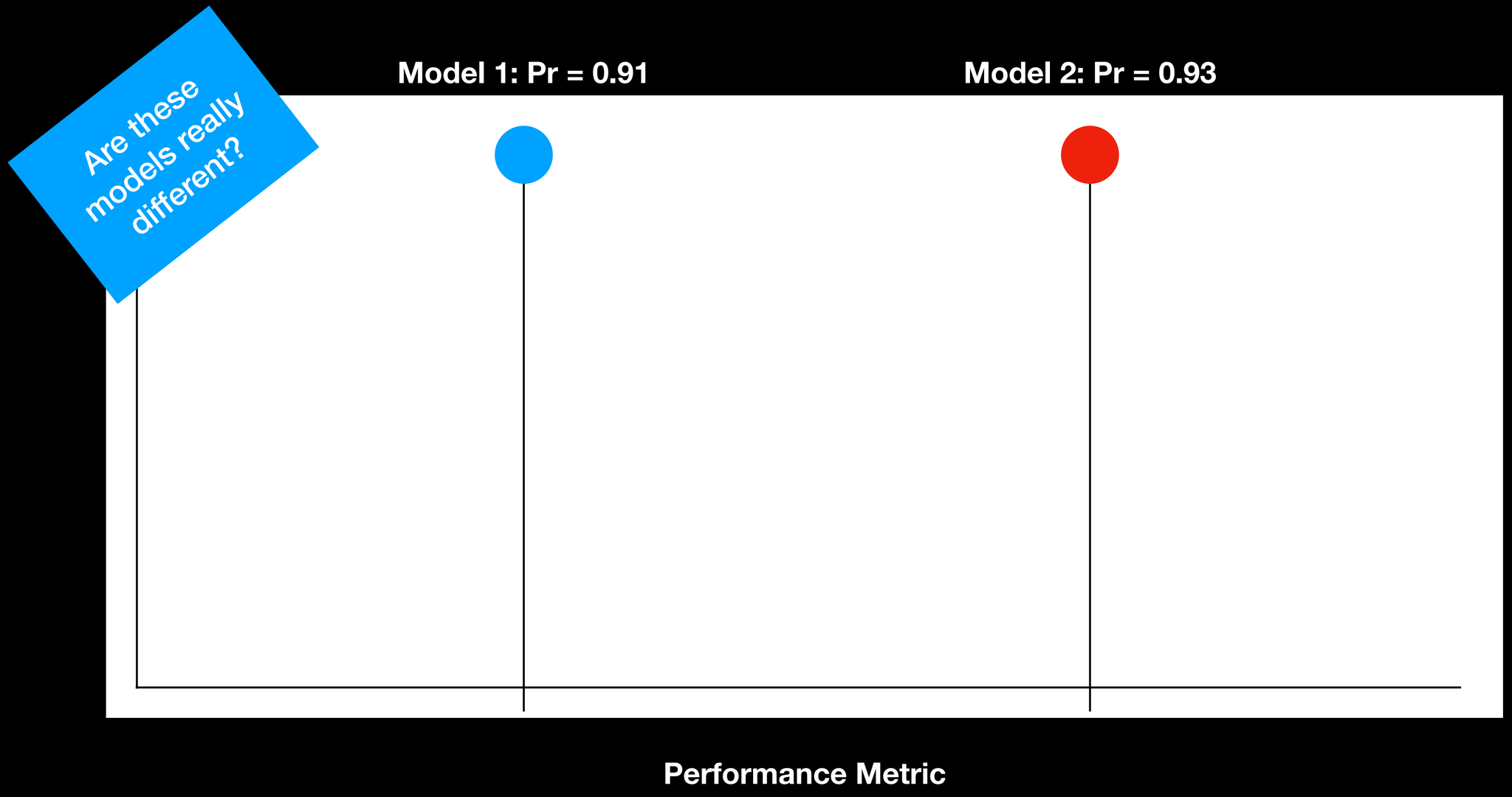


Five-fold Cross Validation

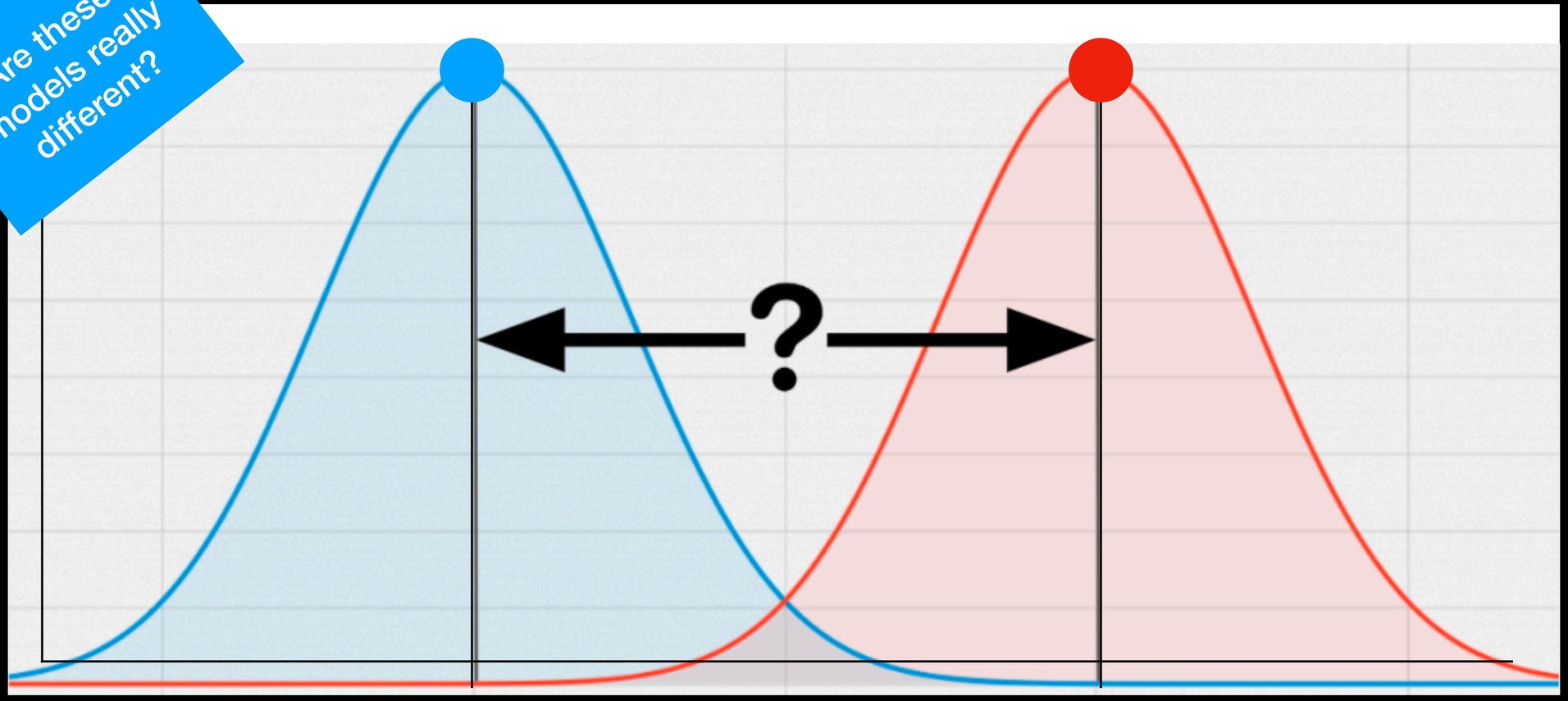


Five-fold Cross Validation

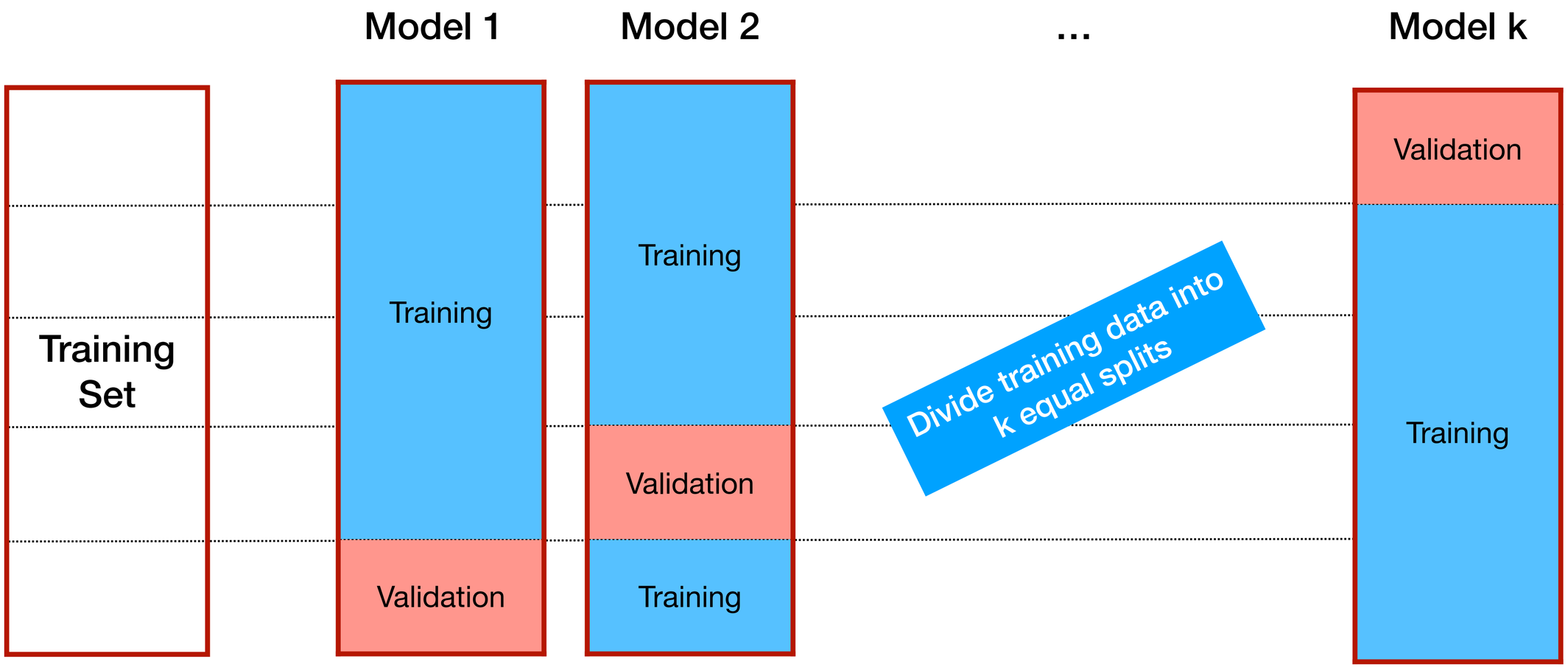



Are these models really different?
Model 1: Pr = 0.91
Model 2: Pr = 0.93
Performance Metric

Are these models really different?



k-fold Cross Validation



Concluding Notes on Model Evaluation and Selection

- Accuracy is often not the right evaluation metric for many real-world machine learning tasks
  - False positives and false negatives may need to be treated very differently.
  - Make sure you understand the needs of your application and choose an evaluation metric that matches your application, user, or business goals.
- Examples of additional evaluation methods include:
  - Learning curve: how much does accuracy (or other metric) change as a function of the amount of training data?
  - Sensitivity analysis: How much does accuracy (or other metric) change as a function of key learning parameter values?

What questions do you have?

Prof. Cody Buntain | @codybuntain | cbuntain@umd.edu
Director, Information Ecosystems Lab