



CSI 436/536 (Spring 2026)

Machine Learning

Lecture 6: Evaluation Criteria

Chong Liu

Department of Computer Science

Feb 9, 2026

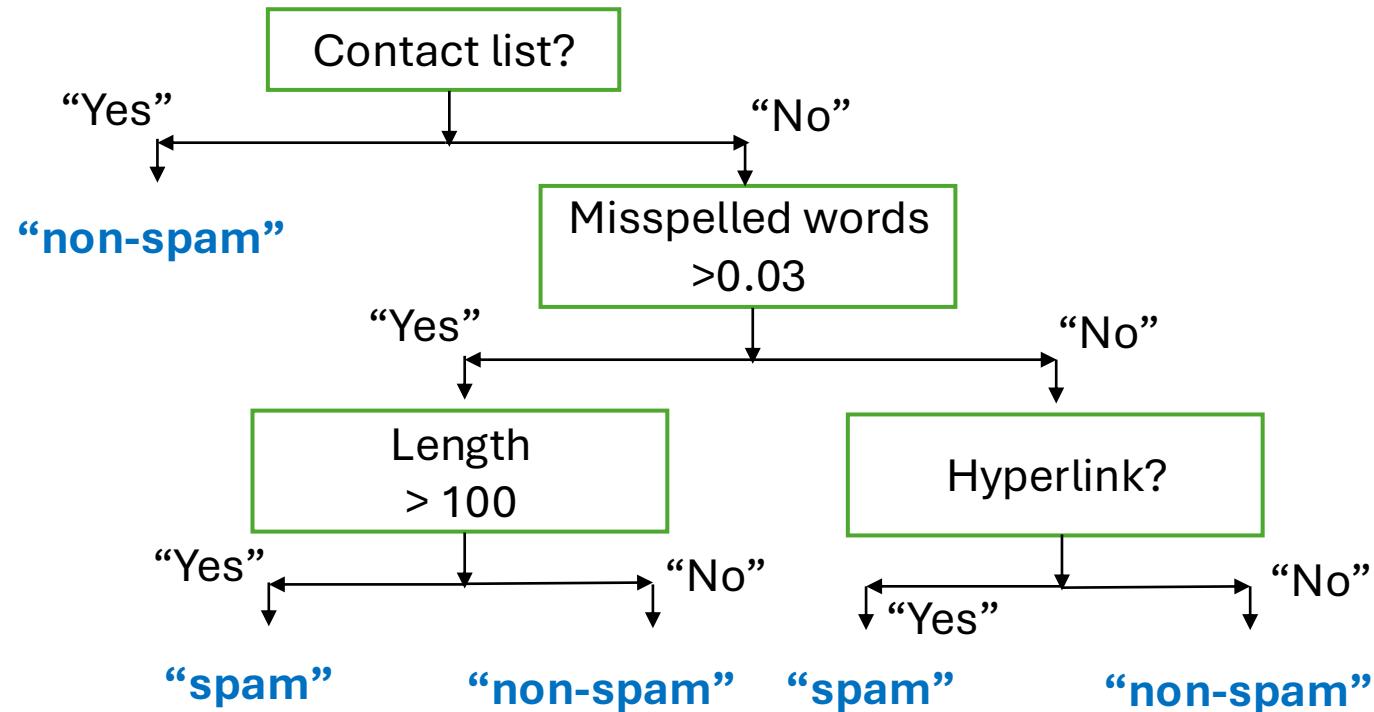
Announcement

- Instructor's office hour cancelled this week.
 - Due to traveling for some invited research talks.
- TA will give a tutorial on LaTeX and Python this Wed.
 - LaTeX for your homework
 - Python for your course project

Recap: elements of machine learning

- Machine learning overview
 - Supervised learning
 - Unsupervised learning
 - Reinforcement learning
- Supervised learning: binary classification
 - Spam filtering
- Feature design and feature extraction
 - In contact list or not
 - Proportion of misspelled words
 - ...
- Decision tree classifier

Recap: Decision tree



Recap: How is a decision tree specified?

- Parameters (built-in parameters of a model)
 - Which feature(s) to use when branching?
 - How to branch? Thresholding? Where to put the threshold?
 - Which label to assign at leaf nodes?
- Hyperparameters (parameters that you can set)
 - Max height of a decision tree?
 - Number of features the tree can use in each branch?

Today

- Linear classifier
- Performance metrics
- Feature transformation

Linear classifiers

- Model:

- Score(x) = $w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4$
- $x_1 = 1$ (has hyperlinks)
- $x_2 = 1$ (on contact list)
- $x_3 = \text{proportion of misspelling}$
- $x_4 = \text{length}$

Indicator function:

$$f(x) = 1(\text{condition}) = \begin{cases} 1, & \text{if condition is true} \\ 0, & \text{if condition is false} \end{cases}$$

Question: why do we need w_0 ?

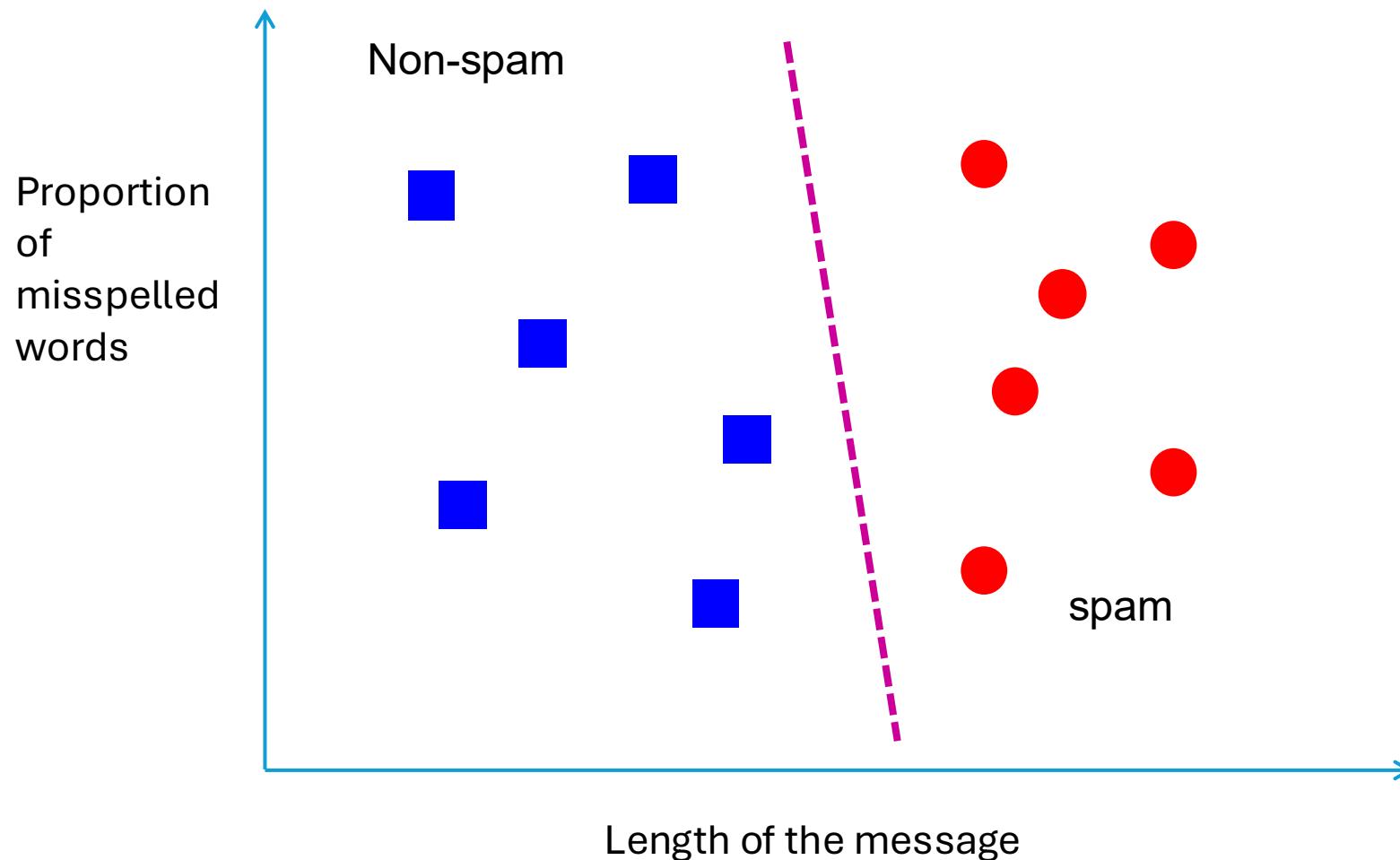
Linear classifiers

- Model:
 - $\text{Score}(x) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4$
- A linear classifier:
 - $$h(x) = \begin{cases} 1, & \text{if } \text{Score}(x) \geq 0 \\ -1, & \text{if } \text{Score}(x) < 0 \end{cases}$$
 - A compact representation:
$$h(x) = \text{sign}(w^T[1; x])$$
- Questions:
 - What are the **parameters** in a linear classifier?
 - Is there any hyperparameter?

Geometric view: Linear classifier is a decision line!

$$\{x | w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 > 0\}$$

The set of all "emails" that will be classified as "Spams"



Family of classifiers: Hypothesis class

- Hypothesis class \mathcal{H}
 - A family of classifiers
 - Also known as “concept class”, “model”, “decision rule book”
 - “Linear classifiers” and “neural networks” are hypothesis classes.
 - Typically we want this family to be large and flexible.
- The task of machine learning:
 - A **selection problem** to find a

$$h \in \mathcal{H}$$

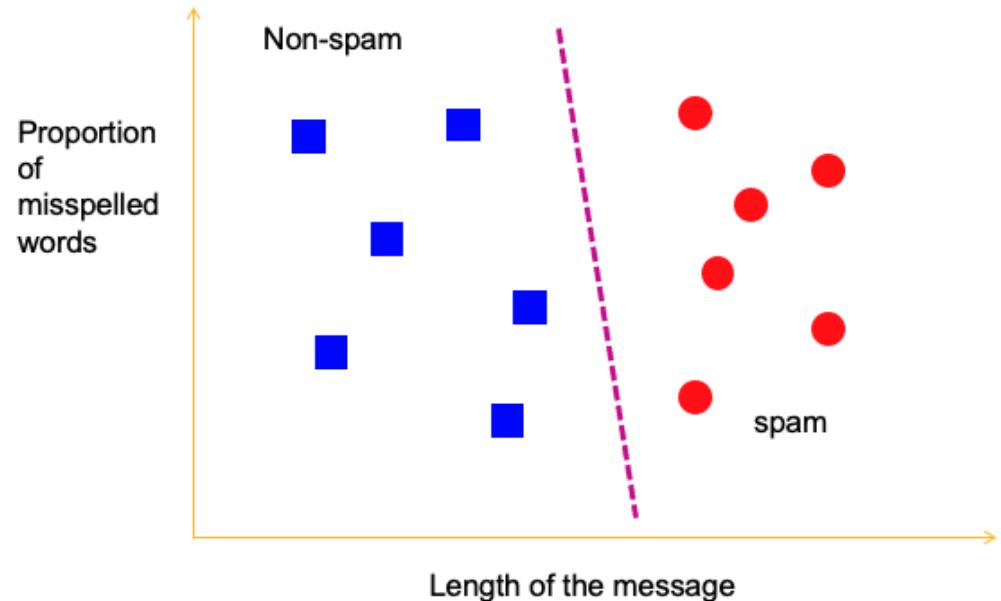
that “**works well**” on this problem.

We will use the following notation to denote a classifier (hypothesis) specified by a specific parameter choice w

$$h_w : \mathcal{X} \rightarrow \mathcal{Y}$$

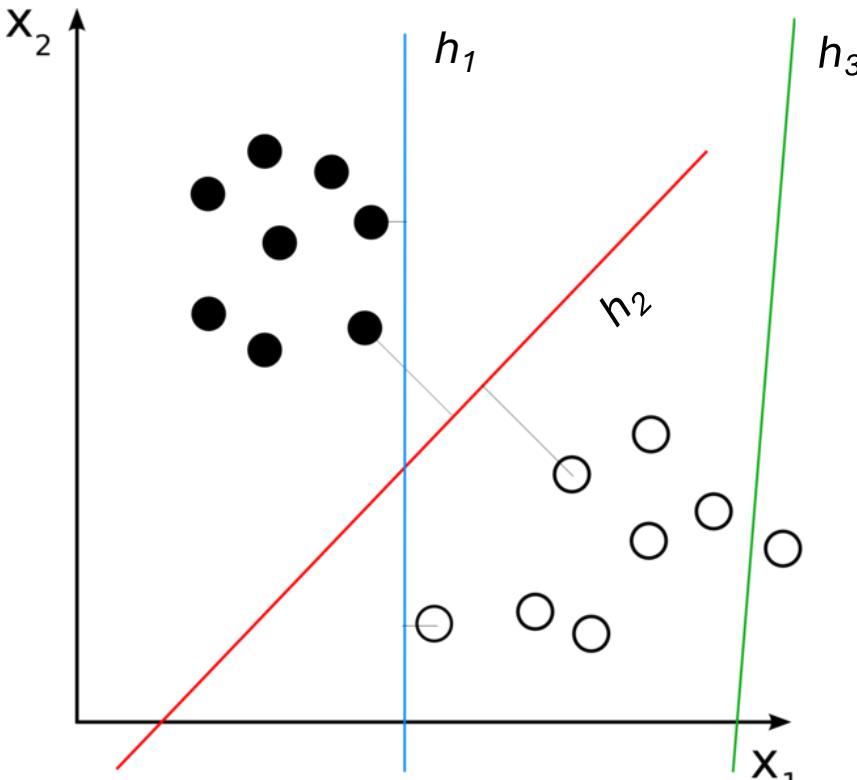
- For any $x \in \mathcal{X}$
 - We can apply this classifier to get its predicted label
$$\hat{y} = h_w(x)$$
 - The prediction doesn't have to be correct. It just need to be valid, i.e.,
$$\hat{y} \in \mathcal{Y}$$

Learning linear classifiers



- Training data:
$$(x_1, y_1), \dots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$$
- There is a clean cut boundary that distinguishes “spams” from “non-spams”.
 - “Linearly separable” problem
 - Learning linear classifier: Finding vector w , such that **the predictions of h_w is consistent with the observed training data.**

Discussion: How can we evaluate a classifier (a spam filter)?



Which is better, h_1 , h_2 , h_3 ? Why?

Confusion matrix for binary classification

		Actual class		Ground truth
		1	0	
		Predicted class		
\hat{y}	1	TP	FP Type I Error	<i>Estimated positive \hat{P}</i>
	0	FN Type II Error	TN	<i>Estimated negative \hat{N}</i>
<i>Positive P</i>		<i>Negative N</i>	TOTAL	

TP – true positives

FP – false positives

TN – true negatives

FN – false negatives

Correct Errors

$$TP + FN = P$$

$$FP + TN = N$$

$$TP + FP = \hat{P}$$

$$FN + TN = \hat{N}$$

$$P + N = TOTAL$$

$$\hat{P} + \hat{N} = TOTAL$$

In-class exercise: confusion matrix

Actual class y ← Ground truth

Classifier output ↓

Predicted class \hat{y}

		Actual class y		
		1	0	
Predicted class \hat{y}	1	TP Type I Error	FP	<i>Estimated positive \hat{P}</i>
	0	FN Type II Error	TN	<i>Estimated negative \hat{N}</i>
		<i>Positive P</i>	<i>Negative N</i>	TOTAL

$$\hat{y} = [1, 1, 1, 1, 0, 0, 0, 1, 1, 1]$$
$$y = [1, 0, 0, 0, 0, 1, 1, 0, 0, 0]$$

Key criteria

- $Accuracy = \frac{TP+TN}{\text{Total}}$
 - correct predictions / total
- $Precision = \frac{TP}{\hat{P}}$
 - Correctly predicted positive observations / total predicted positives
- $Recall = \frac{TP}{P}$
 - Correctly predicted positive observations / all actual positives
- $F1 score = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$

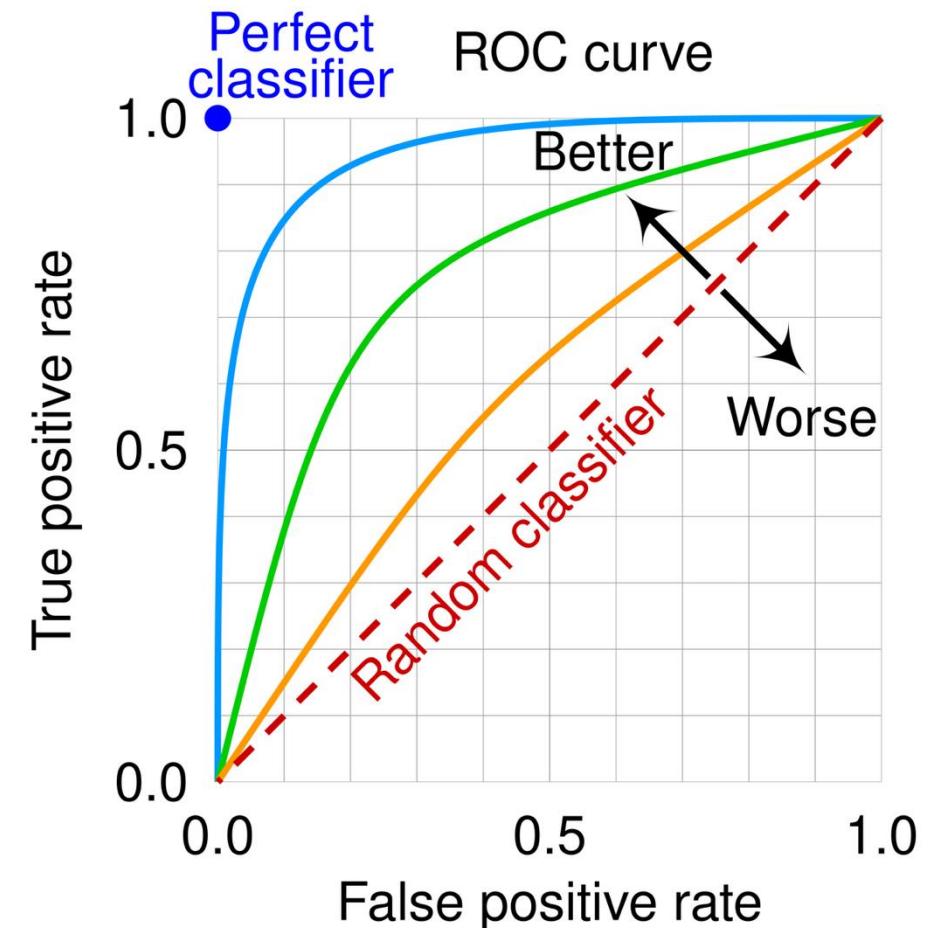
$$\hat{y} = [1,1,1,1,0,0,0,1,1,1]$$

$$y = [1,0,0,0,0,1,1,0,0,0]$$

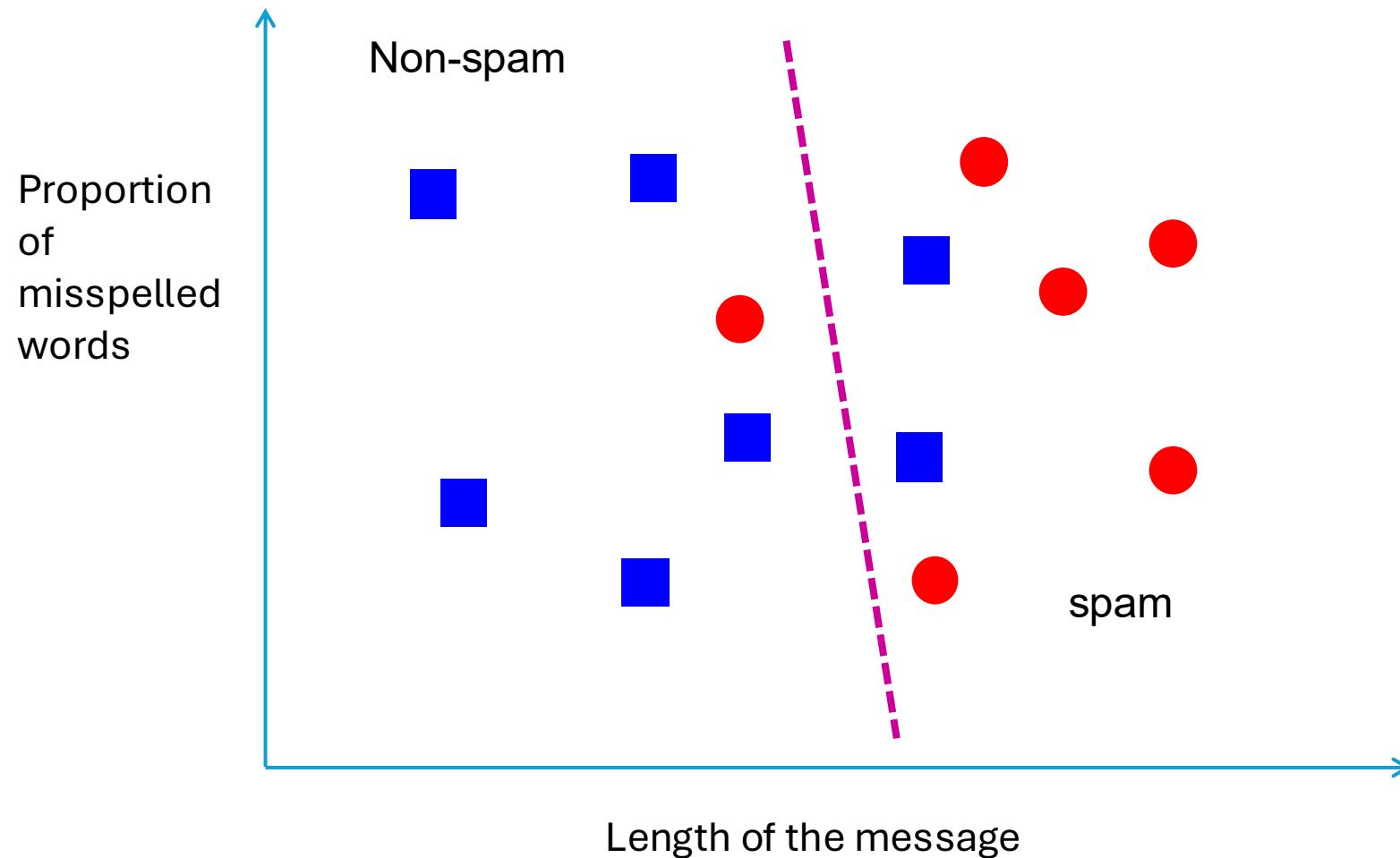
		Actual class		
		1	0	
Predicted class	\hat{y}	1	0	
		TP	FP	<i>Estimated positive \hat{P}</i>
1	0	FN	TN	<i>Estimated negative \hat{N}</i>
		<i>Positive P</i>	<i>Negative N</i>	TOTAL

Key criterion: AUC (Area Under the ROC Curve)

- ROC Cure:
 - Response Operator Characteristic (ROC) curve
 - False positive rate (FPR) = $\frac{FP}{N}$
 - True positive rate (TPR) = $\frac{TP}{P} = Recall$
- AUC:
 - Single number summary of any “**score function**”



In practice: many non-linearly separable case



How to learn a LINEAR classifier in a non-linearly separable case?

- Training data:

$$(x_1, y_1), \dots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$$

- Solving the following optimization problem:

$$\min_{w \in \mathbb{R}^d} \text{Error}(w) = \frac{1}{n} \sum_{i=1}^n \mathbf{1}(h_w(x_i) \neq y_i)$$

- Learning: Find the linear classifier that makes **the smallest number of mistakes** on the training data.