Chapter 8
Evaluation

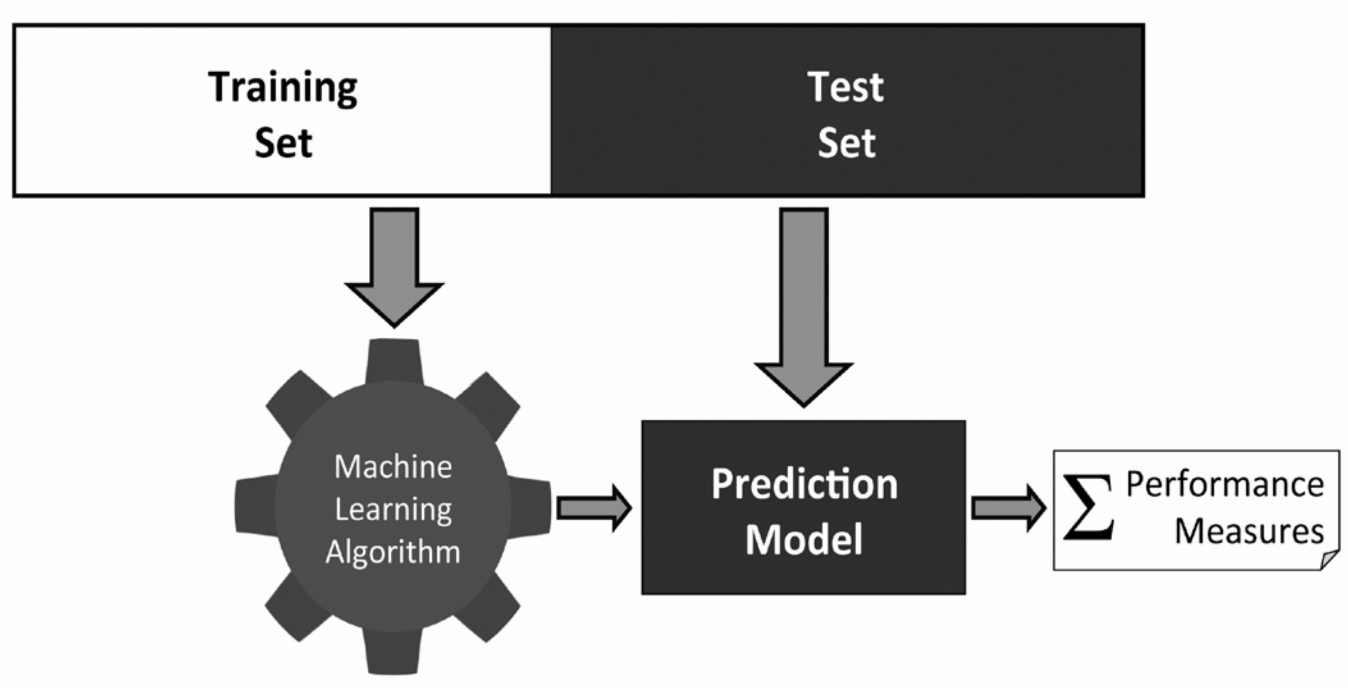
Prof. Chang-Chieh Cheng

Dept. Computer Science

National Chiao Tung University, Taiwan

Evaluation

- Test and evaluation



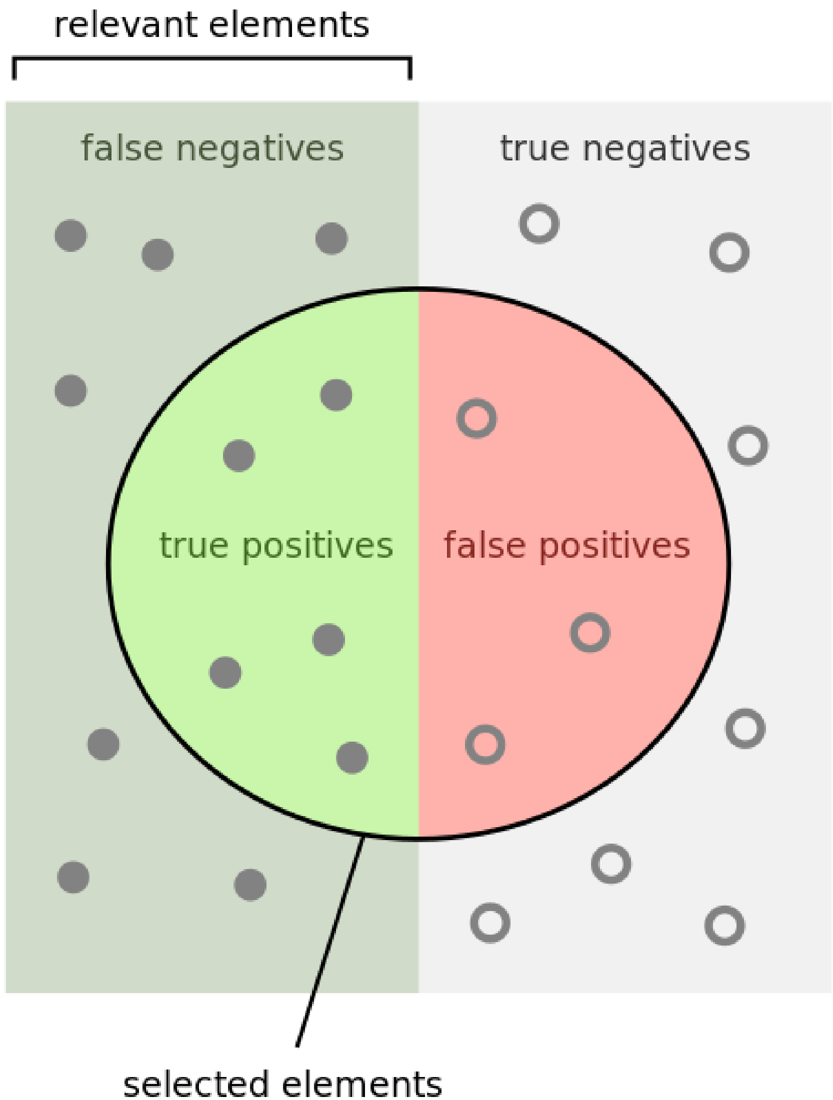
Evaluation

- Basic evaluation

$$misclassification\ rate = \frac{number\ incorrect\ predictions}{total\ predictions}$$

Evaluation

- Four possible outcomes
  - True Positive (TP)
  - True Negative (TN)
  - False Positive (FP)
  - False Negative(FN)



Evaluation

- Confusion matrix

		Prediction	
		positive	negative
Target	positive	*TP*	*FN*
	negative	*FP*	*TN*

Evaluation

- Confusion matrix

ID	Target	Pred.	Outcome	ID	Target	Pred.	Outcome
1	spam	ham	FN	11	ham	ham	TN
2	spam	ham	FN	12	spam	ham	FN
3	ham	ham	TN	13	ham	ham	TN
4	spam	spam	TP	14	ham	ham	TN
5	ham	ham	TN	15	ham	ham	TN
6	spam	spam	TP	16	ham	ham	TN
7	ham	ham	TN	17	ham	spam	FP
8	spam	spam	TP	18	spam	spam	TP
9	spam	spam	TP	19	ham	ham	TN
10	spam	spam	TP	20	ham	spam	FP

		Prediction	
		'spam'	***'ham'***
Target	***'spam'***	6	3
	'ham'	2	9

Evaluation

- Misclassification accuracy

$$\frac{(FP + FN)}{(TP + TN + FP + FN)}$$

$$\frac{(2+3)}{(6+9+2+3)} = 0.25$$

- Classification accuracy

$$\frac{(TP + TN)}{(TP + TN + FP + FN)}$$

$$\frac{(6+9)}{(6+9+2+3)} = 0.75$$

Evaluation

- TP rate (TPR)

$$\frac{TP}{(TP + FN)}$$

- TN rate (TNR)

$$\frac{TN}{(TN + FP)}$$

Evaluation

- FP rate (FPR)

$$\frac{FP}{(TN + FP)}$$

- FN rate (FNR)

$$\frac{FN}{(TP + FN)}$$

Evaluation

- For example

		Prediction	
		'spam'	*'ham'*
Target	*'spam'*	6	3
	'ham'	2	9

$$\begin{matrix} TP & FN \\ FP & TN \end{matrix}$$

$$\begin{aligned} \mathrm{TPR} &= \frac{6}{(6+3)} &= 0.667 \\ \mathrm{TNR} &= \frac{9}{(9+2)} &= 0.818 \\ \mathrm{FPR} &= \frac{2}{(9+2)} &= 0.182 \\ \mathrm{FNR} &= \frac{3}{(6+3)} &= 0.333 \end{aligned}$$

Evaluation

- Precision

$$\frac{TP}{(TP + FP)}$$

- Recall

$$\frac{TP}{(TP + FN)}$$

Evaluation

- For example

		Prediction	
		'spam'	*'ham'*
Target	*'spam'*	6	3
	'ham'	2	9

TP	*FN*
FP	*TN*

$$\text{precision} = \frac{6}{(6+2)} = 0.75$$

$$\text{recall} = \frac{6}{(6+3)} = 0.667$$

Evaluation

- F_1-measure

$$2 \times \frac{(\text{precision} \times \text{recall})}{(\text{precision} + \text{recall})}$$

without TN

$$= \frac{2TP}{2TP + FP + FN}$$

- F_β-measure

$$F_\beta = \frac{(1+\beta^2)TP}{(1+\beta^2)TP + \beta^2 FP + FN}$$

- For example

$$\text{F}_1\text{-measure} = 2 \times \frac{\left(\frac{6}{(6+2)} \times \frac{6}{(6+3)}\right)}{\left(\frac{6}{(6+2)} + \frac{6}{(6+3)}\right)}$$
$$= 0.706$$

Evaluation

- Average class accuracy

$$\frac{1}{|levels(t)|} \sum_{l \in levels(t)} \text{recall}_l$$

- Harmonic average class accuracy

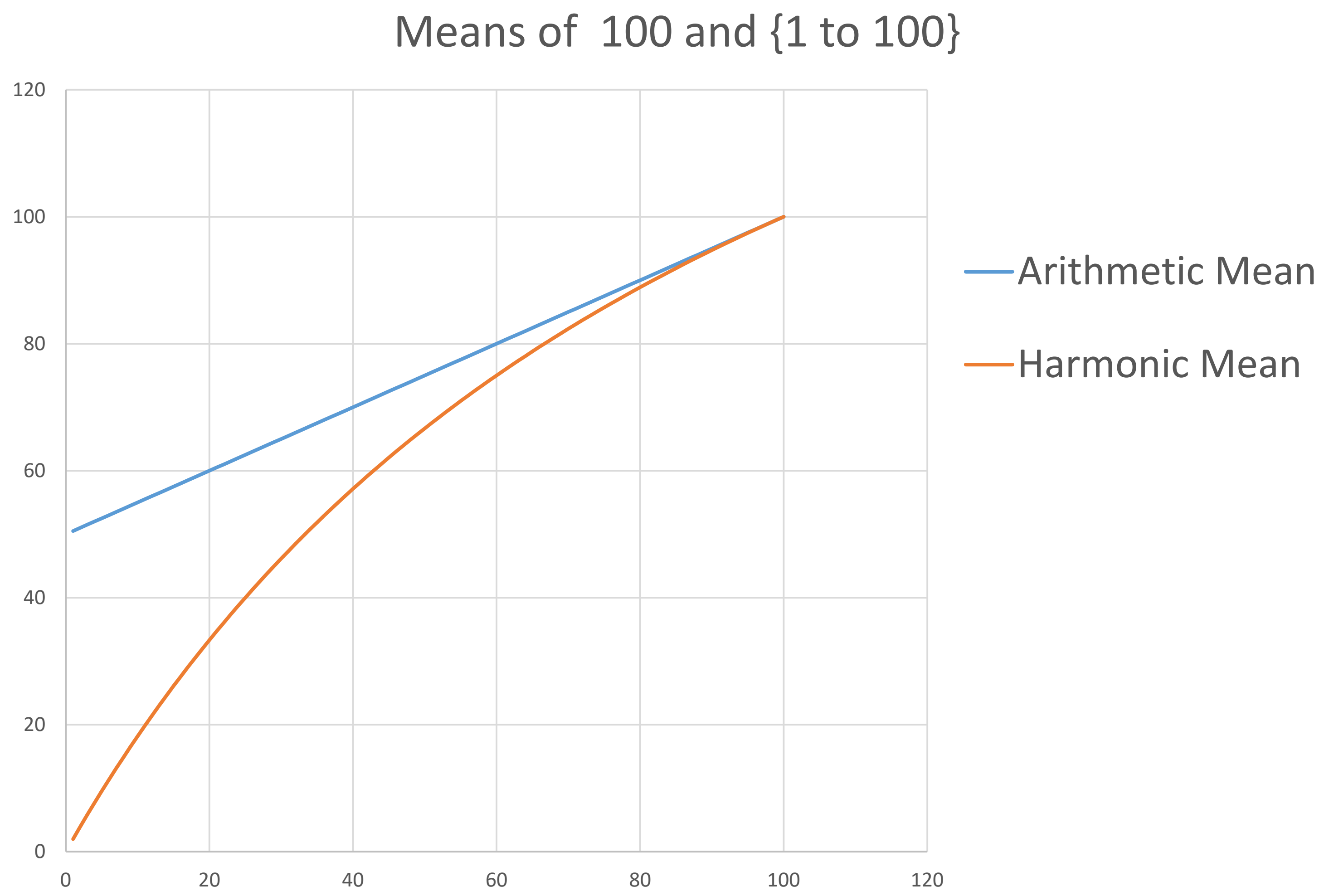
$$\frac{1}{\frac{1}{|levels(t)|} \sum_{l \in levels(t)} \frac{1}{\text{recall}_l}}$$

Harmonic average of *n* numbers:

$$H = \frac{n}{\frac{1}{x_1} + \frac{1}{x_2} + \cdots + \frac{1}{x_n}}$$

Evaluation

- Arithmetic mean and Harmonic mean



Evaluation

- A confusion matrix for a k-NN model trained on a churn prediction problem.

		Prediction 'non-churn'	Prediction 'churn'
Target	'non-churn'	90	0
	'churn'	9	1

$$Recall_{nc} = \frac{90}{90+0} = 1.0$$

$$Recall_{c} = \frac{1}{9+1} = 0.1$$

- A confusion matrix for a naive Bayes model trained on a churn prediction problem.

		Prediction 'non-churn'	Prediction 'churn'
Target	'non-churn'	70	20
	'churn'	2	8

$$Recall_{nc} = \frac{70}{70+20} = 0.778$$

$$Recall_{c} = \frac{8}{2+8} = 0.8$$

Evaluation

- A confusion matrix for a k-NN model trained on a churn prediction problem.

		Prediction	
		'non-churn'	*'churn'*
Target	*'non-churn'*	90	0
	'churn'	9	1

Harmonic average class accuracy

$$= \frac{1}{\frac{1}{2}\left(\frac{1}{1.0} + \frac{1}{0.1}\right)} = 0.182$$

- A confusion matrix for a naive Bayes model trained on a churn prediction problem.

		Prediction	
		'non-churn'	*'churn'*
Target	*'non-churn'*	70	20
	'churn'	2	8

Harmonic average class accuracy

$$= \frac{1}{\frac{1}{2}\left(\frac{1}{0.778} + \frac{1}{0.8}\right)} = 0.789$$

Evaluation

- Profit matrix

		Prediction	
		positive	**negative**
Target	**positive**	TP_{Profit}	FN_{Profit}
	negative	FP_{Profit}	TN_{Profit}

Evaluation

- Example
  - The **profit matrix** for the pay-day loan credit scoring problem.

		Prediction	
		'good'	*'bad'*
Target	*'good'*	140	–140
	'bad'	–700	0

Evaluation

- Example
  - The confusion matrix for a k-NN model trained on the pay-day loan credit scoring problem.

		Prediction	
		'good'	*'bad'*
Target	*'good'*	57	3
	'bad'	10	30

$$\text{average class accuracy}_{HM} = 83.824\%$$

  - The confusion matrix for a decision tree model trained on the pay-day loan credit scoring problem

		Prediction	
		'good'	*'bad'*
Target	*'good'*	43	17
	'bad'	3	37

$$\text{average class accuracy}_{HM} = 80.761\%$$

Evaluation

- Overall profit for the k-NN model

		Prediction 'good'	Prediction 'bad'
Target	'good'	7 980	−420
	'bad'	−7 000	0
	Profit		**560**

- Overall profit for the decision tree model

		Prediction 'good'	Prediction 'bad'
Target	'good'	6 020	−2 380
	'bad'	−2 100	0
	Profit		**1 540**

Evaluation

- Multinomial targets

$$precision(l) = \frac{TP(l)}{TP(l) + FP(l)}$$

$$recall(l) = \frac{TP(l)}{TP(l) + FN(l)}$$

  - where l is a target level

Evaluation

- Multinomial targets
  - Example: bacterial species identification

ID	Target	Prediction	ID	Target	Prediction
1	durionis	fructosus	16	ficulneus	ficulneus
2	ficulneus	fructosus	17	ficulneus	ficulneus
3	fructosus	fructosus	18	fructosus	fructosus
4	ficulneus	ficulneus	19	durionis	durionis
5	durionis	durionis	20	fructosus	fructosus
6	pseudo.	pseudo.	21	fructosus	fructosus
7	durionis	fructosus	22	durionis	durionis
8	ficulneus	ficulneus	23	fructosus	fructosus
9	pseudo.	pseudo.	24	pseudo.	fructosus
10	pseudo.	fructosus	25	durionis	durionis
11	fructosus	fructosus	26	pseudo.	pseudo.
12	ficulneus	ficulneus	27	fructosus	fructosus
13	durionis	durionis	28	ficulneus	ficulneus
14	fructosus	fructosus	29	fructosus	fructosus
15	fructosus	ficulneus	30	fructosus	fructosus

Evaluation

- Multinomial targets
  - Example: bacterial species identification

		Prediction				Recall
		'durionis'	*'ficulneus'*	*'fructosus'*	*'pseudo.'*	
Target	*'durionis'*	5	0	2	0	0.714
	'ficulneus'	0	6	1	0	0.857
	'fructosus'	0	1	10	0	0.909
	'pseudo.'	0	0	2	3	0.600
	Precision	1.000	0.857	0.667	1.000	

  - Harmonic average class accuracy

$$\frac{1}{\frac{1}{4}\left(\frac{1}{0.714}+\frac{1}{0.857}+\frac{1}{0.909}+\frac{1}{0.600}\right)} = \frac{1}{1.333} = 75.000\%$$

Evaluation

- Continuous targets

$$\text{sum of squared errors} = \frac{1}{2}\sum_{i=1}^{n}(t_i - \mathbb{M}(\mathbf{d}_i))^2$$

$$\text{mean squared error} = \frac{\sum_{i=1}^{n}(t_i - \mathbb{M}(\mathbf{d}_i))^2}{n}$$

$$\text{root mean squared error} = \sqrt{\frac{\sum_{i=1}^{n}(t_i - \mathbb{M}(\mathbf{d}_i))^2}{n}}$$

$$\text{mean absolute error} = \frac{\sum_{i=1}^{n} abs(t_i - \mathbb{M}(\mathbf{d}_i))}{n}$$

Evaluation

- Continuous targets

$$R^2 = 1 - \frac{\text{sum of squared errors}}{\text{total sum of squares}}$$

$$\text{total sum of squares} = \frac{1}{2}\sum_{i=1}^{n}(t_i - \bar{t})^2$$

Validation

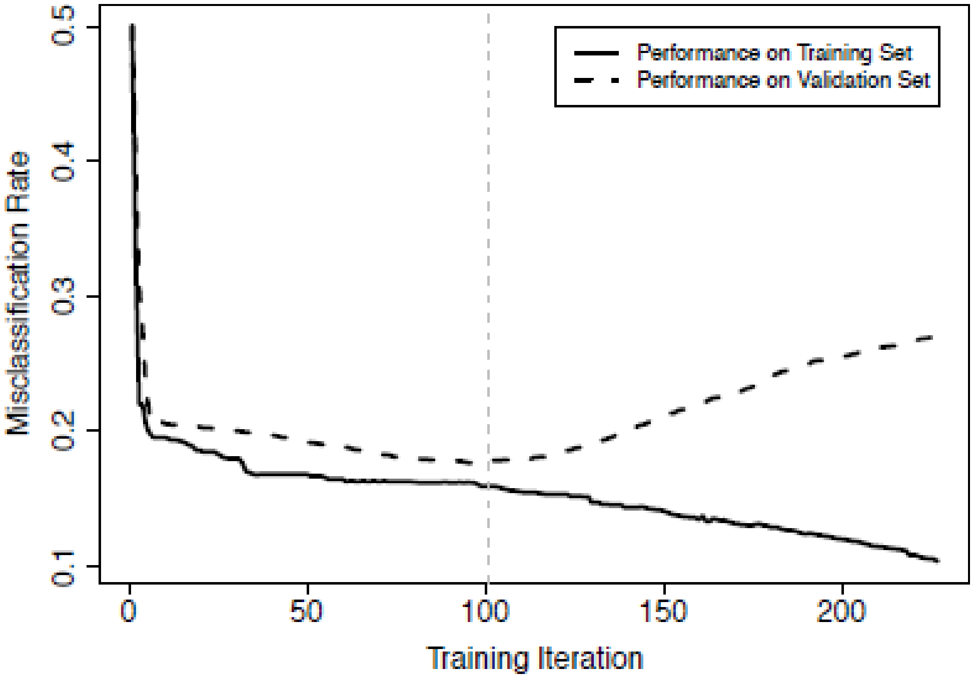
- Hold-out sampling
  - Divide the full data into training, validation, and test sets.

(a) A 50:20:30 split

(b) A 40:20:40 split

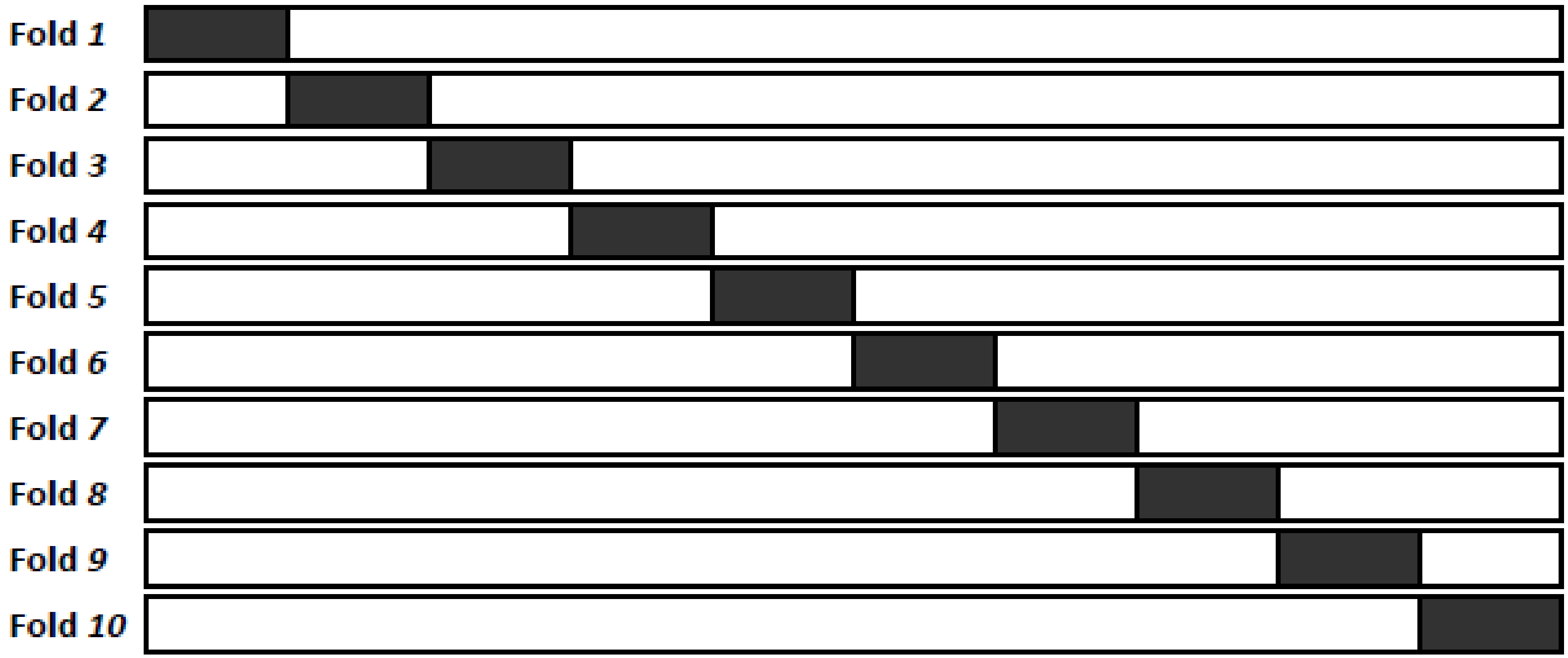
Validation

- Validation set
  - To tune the parameters of a model
  - To avoid overfitting in iterative machine learning algorithms.
- Test set
  - only to assess the performance of a model.



Validation

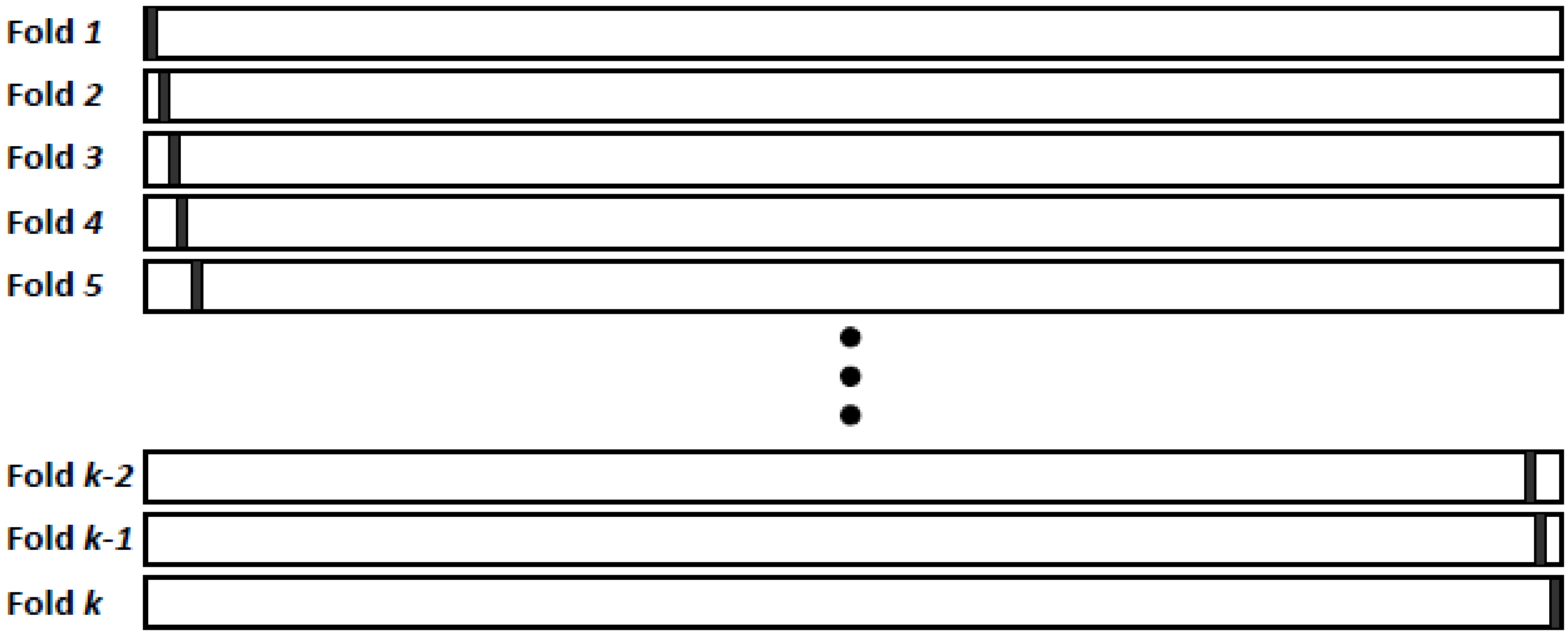
- K-ford cross validation



Black rectangles indicate test data,
and white spaces indicate training data.

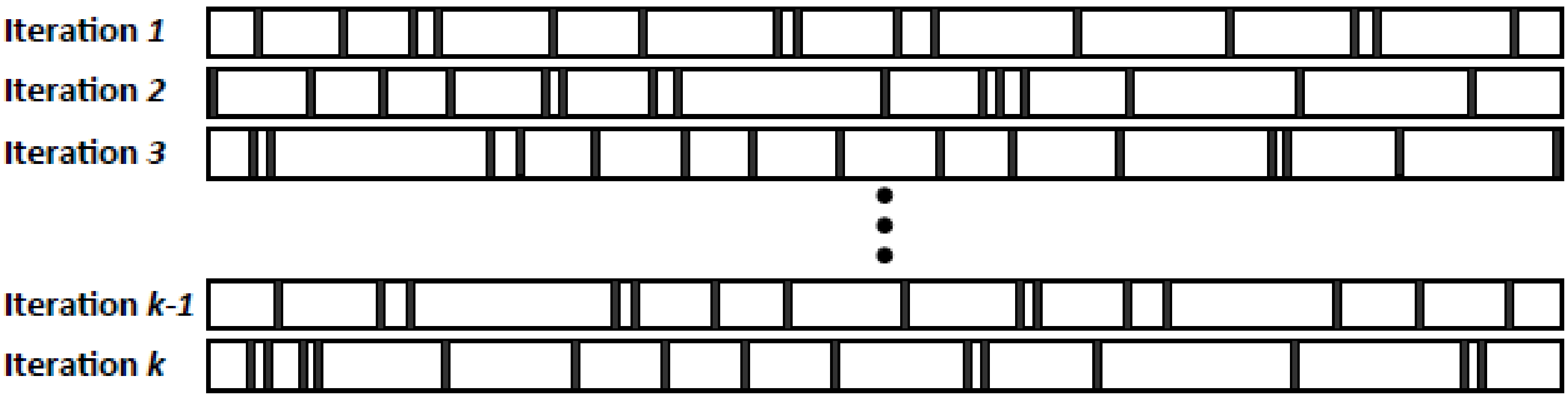
Validation

- Leave-one-out cross validation



Validation

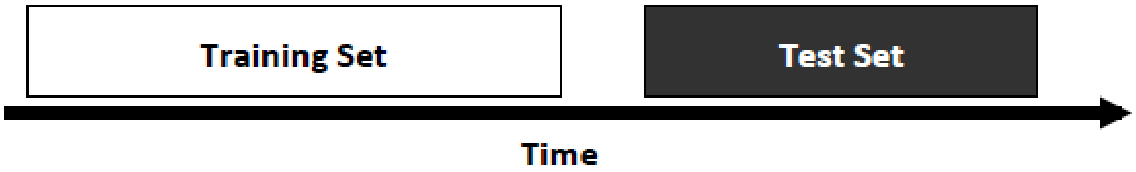
- ϵ0 bootstrap process
  - **bootstrapping**: a self-starting process
  - k iterations
  - Each iteration randomly select *m* instances as training set



Black rectangles indicate test data,
and white spaces indicate training data.

Validation

- Notice the out-of-time sampling



Performance

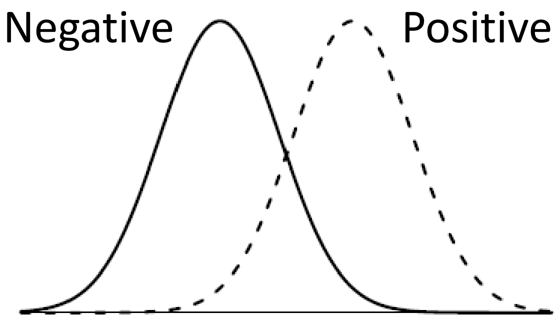
- Prediction scores
  - all classification prediction models return a score
  - score ≧ threshold ➔Positive
  - otherwise ➔Negative

ID	Target	Pred-iction	Score	Out-come	ID	Target	Pred-iction	Score	Out-come
7	ham	ham	0.001	TN	5	ham	ham	0.302	TN
11	ham	ham	0.003	TN	14	ham	ham	0.348	TN
15	ham	ham	0.059	TN	17	ham	spam	0.657	FP
13	ham	ham	0.064	TN	8	spam	spam	0.676	TP
19	ham	ham	0.094	TN	6	spam	spam	0.719	TP
12	spam	ham	0.160	FN	10	spam	spam	0.781	TP
2	spam	ham	0.184	FN	18	spam	spam	0.833	TP
3	ham	ham	0.226	TN	20	ham	spam	0.877	FP
16	ham	ham	0.246	TN	9	spam	spam	0.960	TP
1	spam	ham	0.293	FN	4	spam	spam	0.963	TP

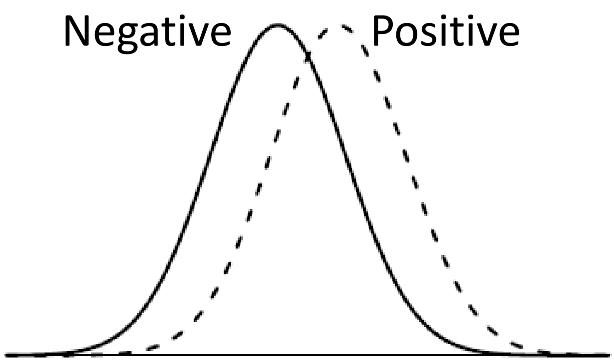
spam is positive and **ham is negative** in this case
Target is spam and prediction is also spam➔TP
Target is spam and but prediction is ham➔FN
Target is ham and prediction is also ham➔TN
Target is ham and but prediction is spam ➔FP

Performance

- Prediction scores
  - How well the distributions of scores produced by the model for different target levels are separated?
  - Which model is better?
    - Model 1



    - Model 2



Performance

- Prediction scores
  - Threshold increases TPR decreases
  - TP rate (TPR) = TP / (TP + FN)
    - Threshold = 0.0 ➔ Every thing is positive ➔ FN = 0
  - Threshold increases TNR increases
  - TN rate (TNR) = TN / (TN + FP)
    - Threshold = 0.0 ➔ No negative ➔ TN = 0

Performance

- Receiver operating characteristic index, ROC index
- Receiver operating characteristic curve, ROC curve
- Example:

(a) Threshold: 0.75

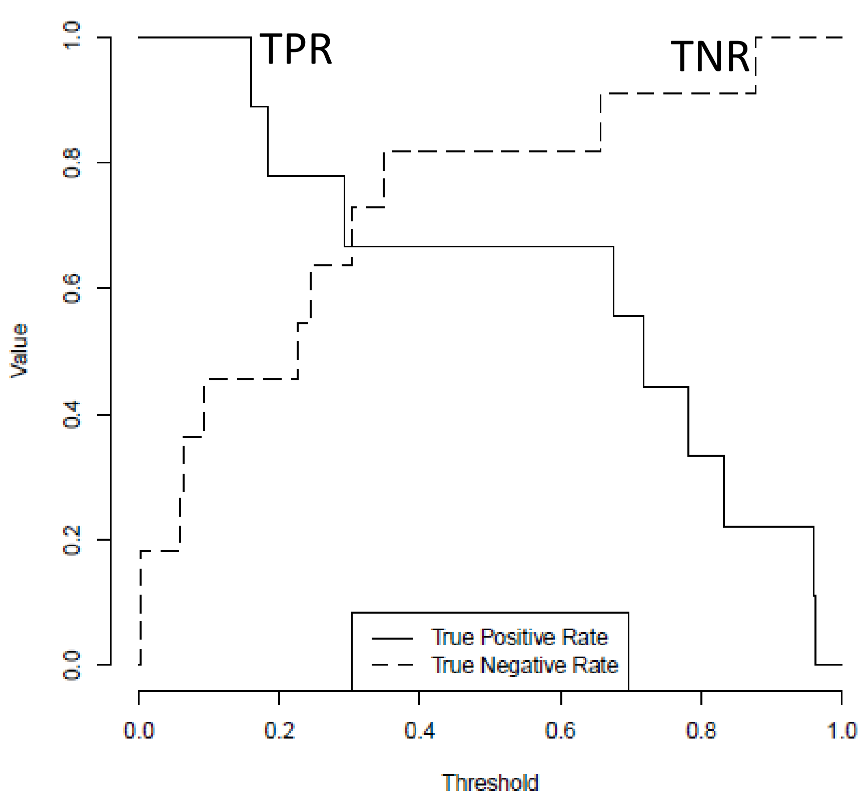
		Prediction *'spam'*	Prediction *'ham'*
Target	*'spam'*	4	4
	'ham'	2	10

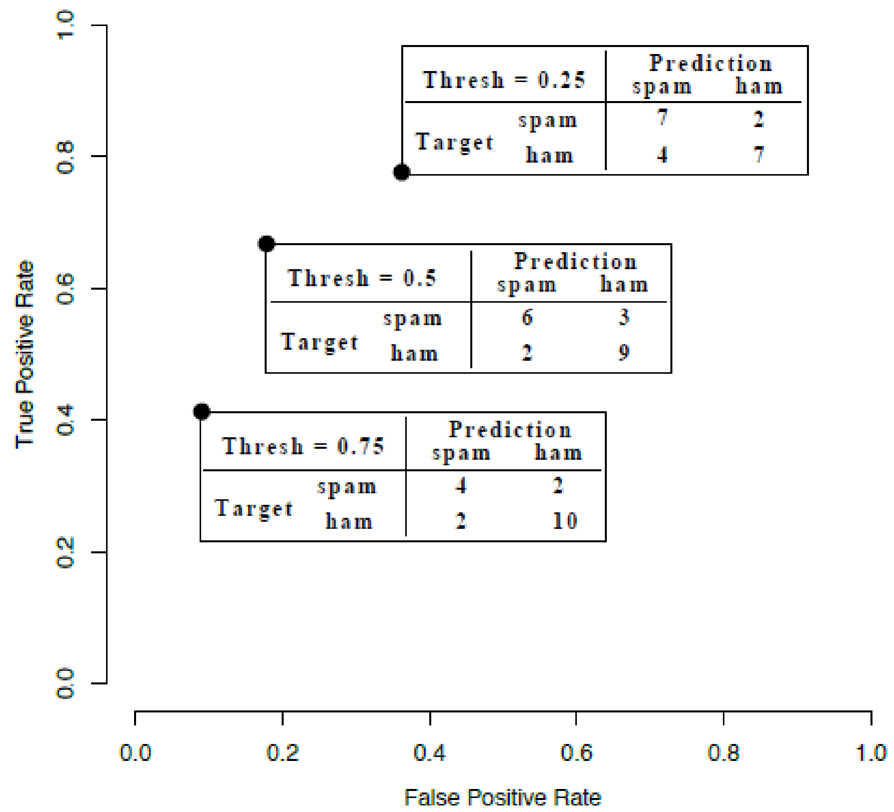
(b) Threshold: 0.25

		Prediction *'spam'*	Prediction *'ham'*
Target	*'spam'*	7	2
	'ham'	4	7

Performance

- ROC curve





TP rate (TPR) = TP / (TP + FN)
TN rate (*TNR*) = TN / (TN + FP)

Performance

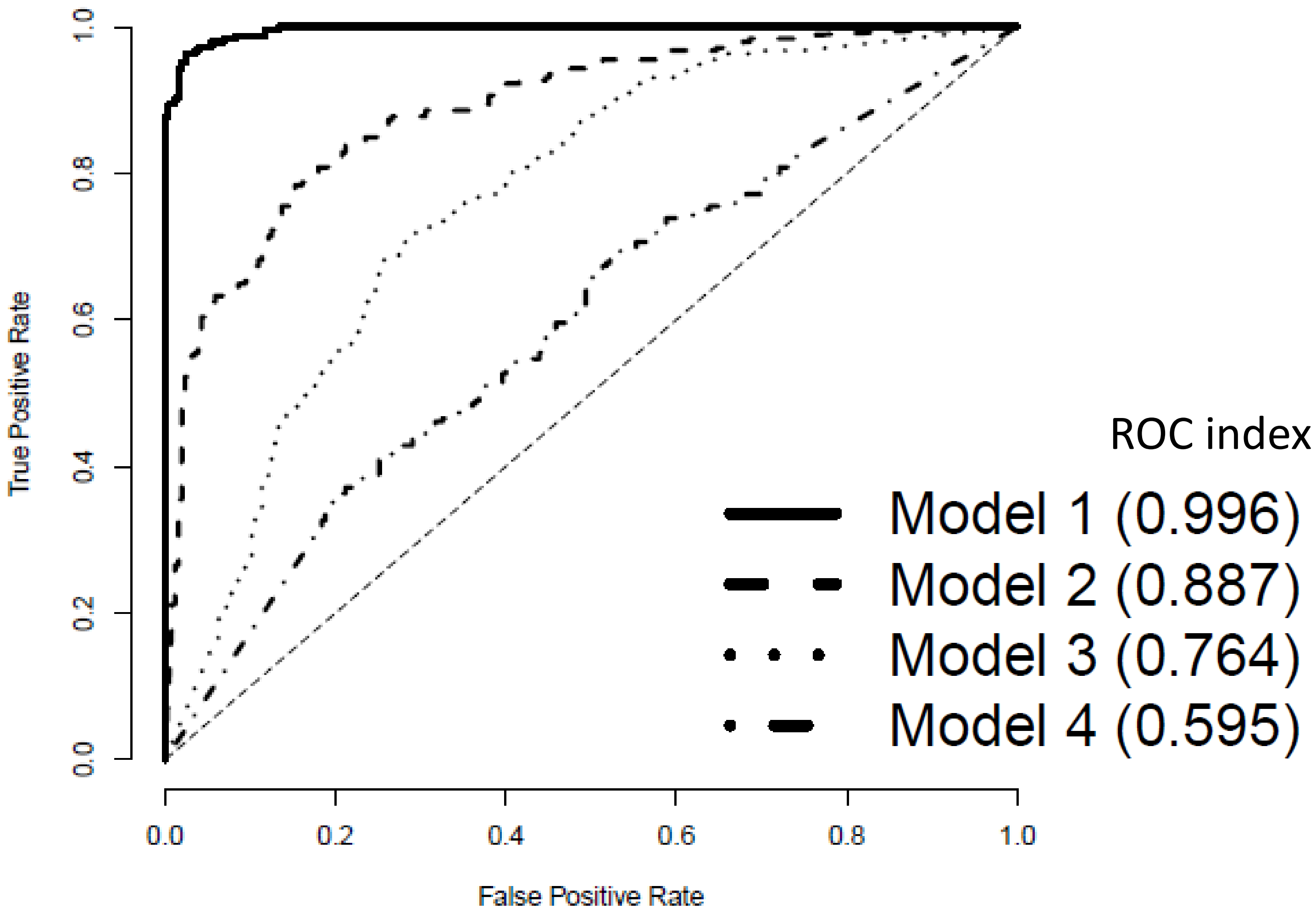
- ROC index
  - Given a set of thresholds $T = \{t_1, t_2, \ldots, t_m\}$

$$R = \sum_{i=2}^{m} \frac{(FPR(t_i) - FPR(t_{i-1}))(TPR(t_i) + TPR(t_{i-1}))}{2}$$

  - R is above 0.7 that indicates a strong model; otherwise, weak model

Performance

- ROC curve



Performance

- Kolmogorov-Smirnov statistic (K-S statistic)

$$CP(\mathit{positive}, ps) = \frac{\text{num positive test instances with score} \leq ps}{\text{num positive test instances}}$$

$$CP(\mathit{negative}, ps) = \frac{\text{num negative test instances with score} \leq ps}{\text{num negative test instances}}$$

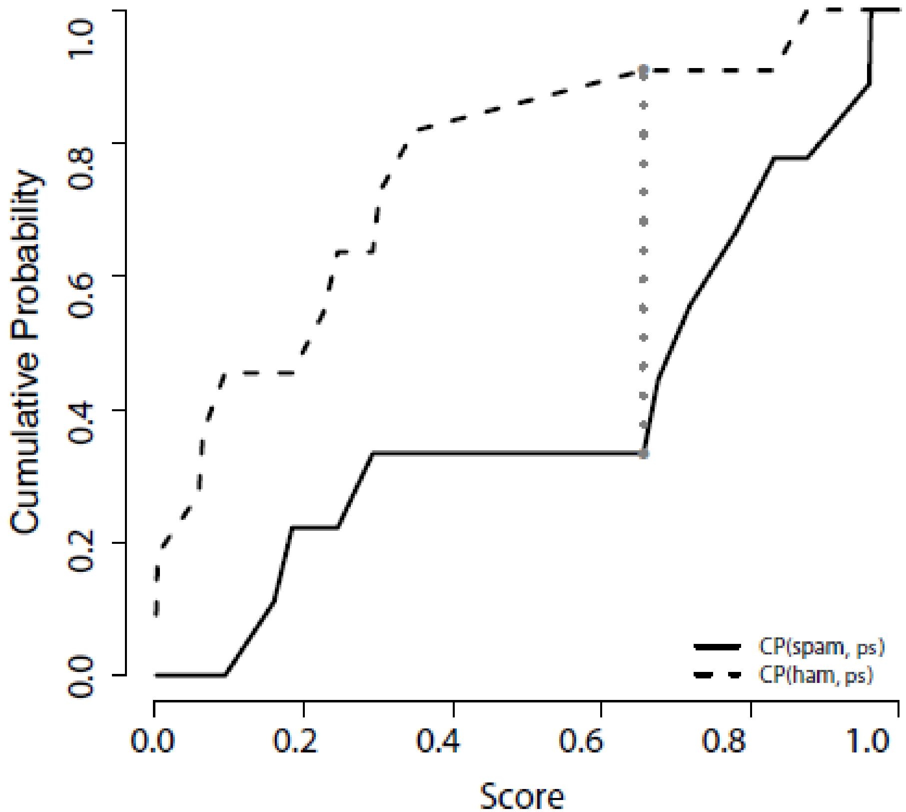
  - where *ps* is prediction score

Performance

- Kolmogorov-Smirnov statistic (K-S statistic)

$$\text{K-S} = \max_{ps}\left(CP(positive, ps) - CP(negative, ps)\right)$$

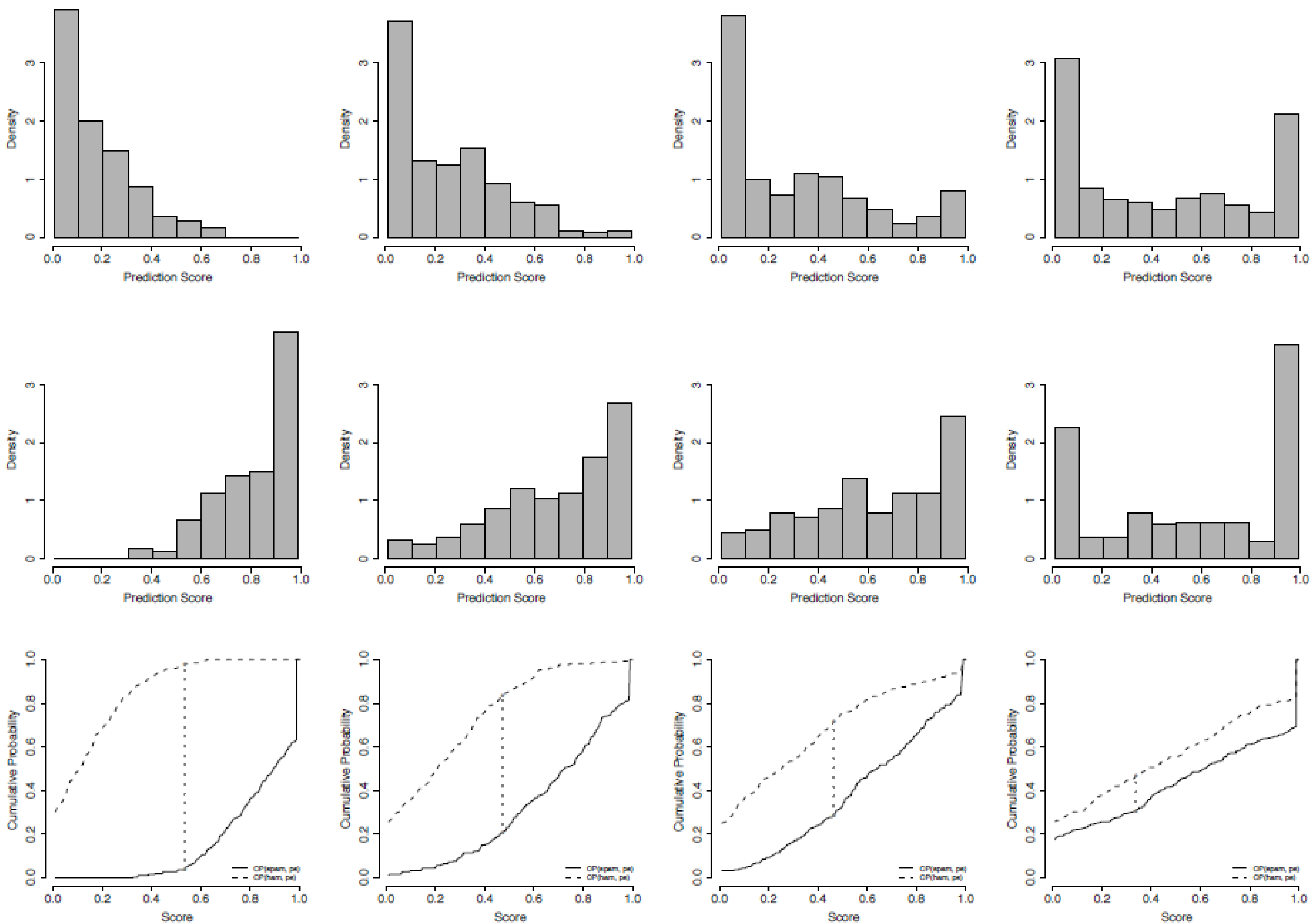
  - Higher K-S indicate better model



Performance

- Kolmogorov-Smirnov statistic (K-S statistic)

(a) Model 1 (b) Model 2 (c) Model 3 (d) Model 4

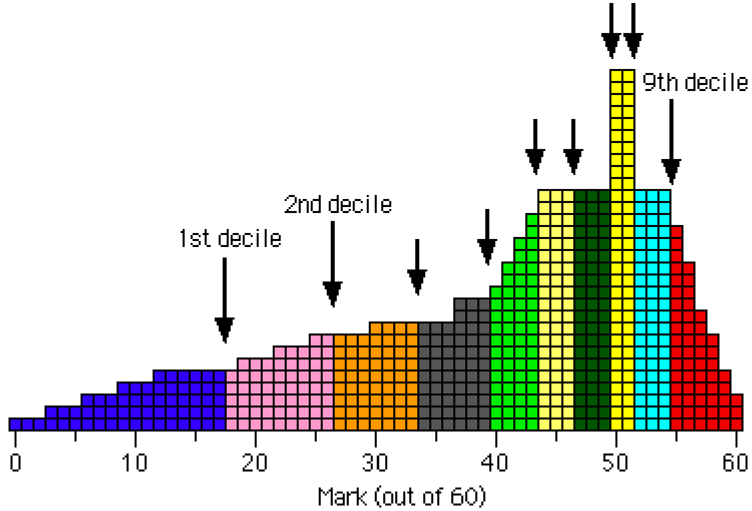


Performance

- Gain

$$\text{Gain}(dec) = \frac{\text{num positive test instances in decile } dec}{\text{num positive test instances}}$$

- a **decile** is any of the nine values that divide the sorted data into ten equal parts



Separating 10 parts by 9 values

Performance

- Sorted by prediction score
- 10 deciles separating

Decile	ID	Target	Prediction	Score	Outcom
1^{st}	9	spam	spam	0.960	TP
	4	spam	spam	0.963	TP
2^{nd}	18	spam	spam	0.833	TP
	20	ham	spam	0.877	FP
3^{rd}	6	spam	spam	0.719	TP
	10	spam	spam	0.781	TP
4^{th}	17	ham	spam	0.657	FP
	8	spam	spam	0.676	TP
5^{th}	5	ham	ham	0.302	TN
	14	ham	ham	0.348	TN
6^{th}	16	ham	ham	0.246	TN
	1	spam	ham	0.293	FN
7^{th}	2	spam	ham	0.184	FN
	3	ham	ham	0.226	TN
8^{th}	19	ham	ham	0.094	TN
	12	spam	ham	0.160	FN
9^{th}	15	ham	ham	0.059	TN
	13	ham	ham	0.064	TN
10^{th}	7	ham	ham	0.001	TN
	11	ham	ham	0.003	TN

Performance

- Example: **gain**, **cumulative gain**, **lift**, and **cumulative lift**

Decile	Positive ('spam') Count	Negative ('ham') Count	Gain	Cum. Gain	Lift	Cum. Lift
1^{st}	2	0	0.222	0.222	2.222	2.222
2^{nd}	1	1	0.111	0.333	1.111	1.667
3^{rd}	2	0	0.222	0.556	2.222	1.852
4^{th}	1	1	0.111	0.667	1.111	1.667
5^{th}	0	2	0.000	0.667	0.000	1.333
6^{th}	1	1	0.111	0.778	1.111	1.296
7^{th}	1	1	0.111	0.889	1.111	1.270
8^{th}	1 $\Sigma = 9$	1 $\Sigma = 7$	0.111	1.000	1.111	1.250
9^{th}	0	2	0.000	1.000	0.000	1.111
10^{th}	0	2	0.000	1.000	0.000	1.000

#Positive: 9, (9/20)
#Negative: 11, (11/20)

=(1/2)/(9/20)

=(9/(9+7))/(9/20)

Performance

$$\text{Cumulative gain}(\mathit{dec}) = \frac{\text{num positive test instances in all deciles up to } \mathit{dec}}{\text{num positive test instances}}$$

$$\text{Lift}(\mathit{dec}) = \frac{\text{\% of positive test instances in decile } \mathit{dec}}{\text{\% of positive test instances}}$$

$$\text{Cumulative lift}(\mathit{dec}) = \frac{\text{\% of positive instances in all deciles up to } \mathit{dec}}{\text{\% of positive test instances}}$$

Performance