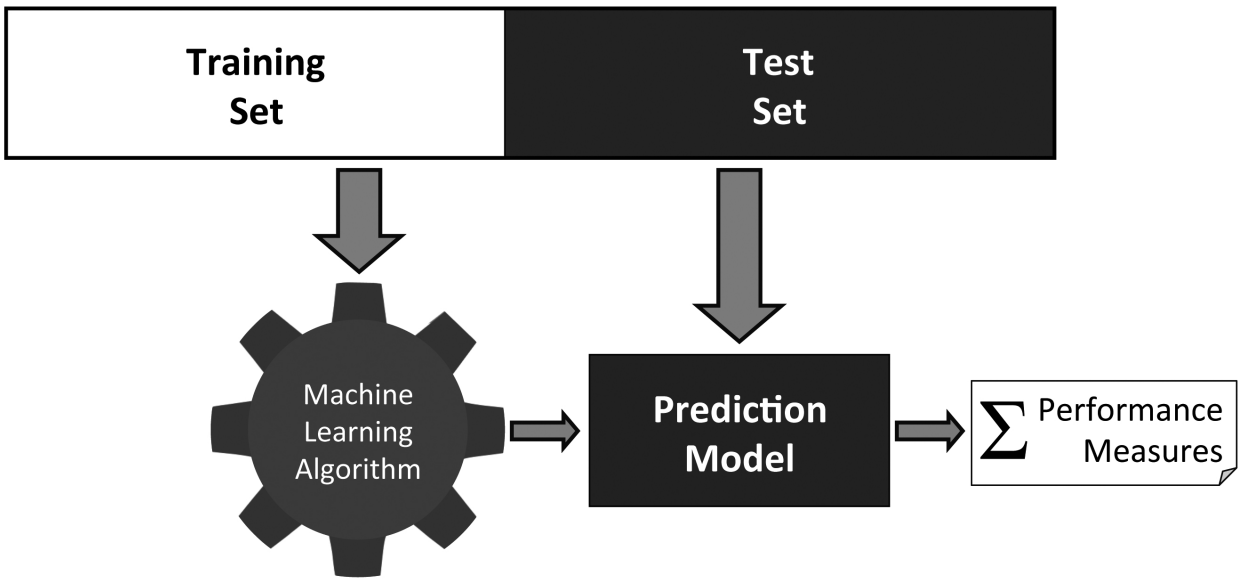
Evaluation

- Big Idea
- Standard Approach: Measuring Misclassification Rate on a Hold-out Test Set
- Designing Evaluation Experiments
- Performance Measures: Categorical Targets
- Performance Measures: Prediction Scores
- Performance Measures: Continuous Targets

Big Idea

- The purpose of evaluation is threefold:
  1. to determine which model is the most suitable for a task
  2. to estimate how the model will perform
  3. to convince users that the model will meet their needs

Standard Approach: Measuring Misclassification Rate on a Hold-out Test Set



The process of building and evaluating a model using a **hold-out test set**.

A sample test set with model predictions.

ID	Target	Pred.	Outcome
1	spam	ham	FN
2	spam	ham	FN
3	ham	ham	TN
4	spam	spam	TP
5	ham	ham	TN
6	spam	spam	TP
7	ham	ham	TN
8	spam	spam	TP
9	spam	spam	TP
10	spam	spam	TP

ID	Target	Pred.	Outcome
11	ham	ham	TN
12	spam	ham	FN
13	ham	ham	TN
14	ham	ham	TN
15	ham	ham	TN
16	ham	ham	TN
17	ham	spam	FP
18	spam	spam	TP
19	ham	ham	TN
20	ham	spam	FP

$$\text{misclassification rate} = \frac{\text{number incorrect predictions}}{\text{total predictions}} \tag{1}$$

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

- For binary prediction problems there are 4 possible outcomes:
  1. True Positive (TP)
  2. True Negative (TN)
  3. False Positive (FP)
  4. False Negative (FN)

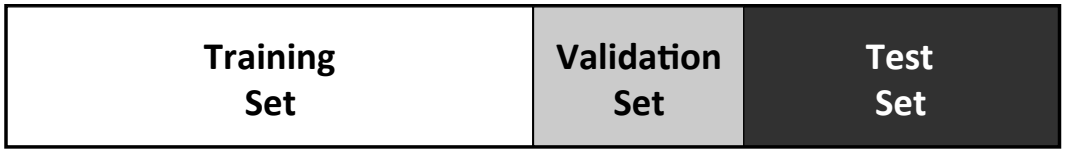
		Prediction	
		positive	**negative**
Target	**positive**	*TP*	*FN*
	negative	*FP*	*TN*

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

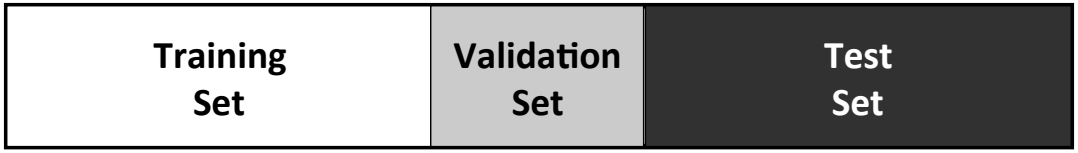
$$\text{classification accuracy} = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{2}$$

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

Designing Evaluation Experiments

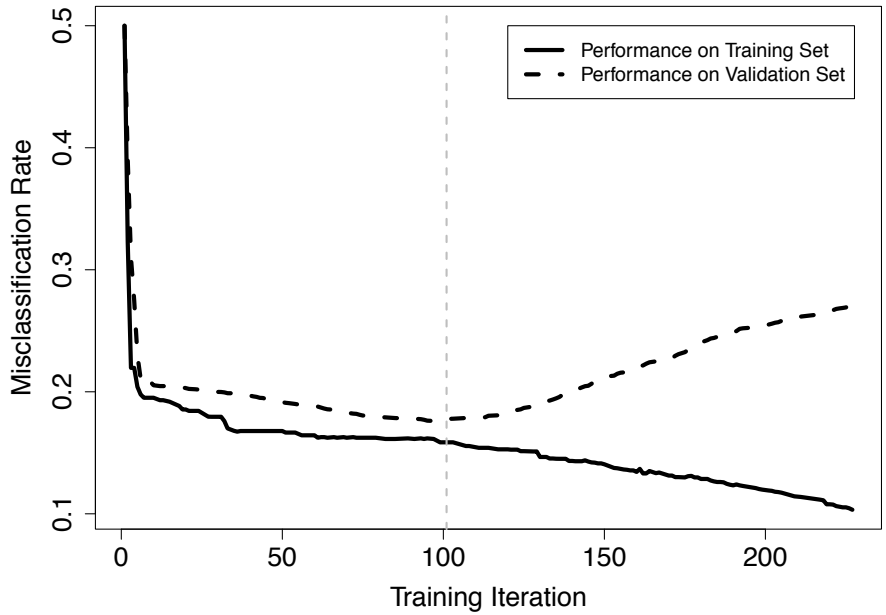


(a) A 50:20:30 split



(b) A 40:20:40 split

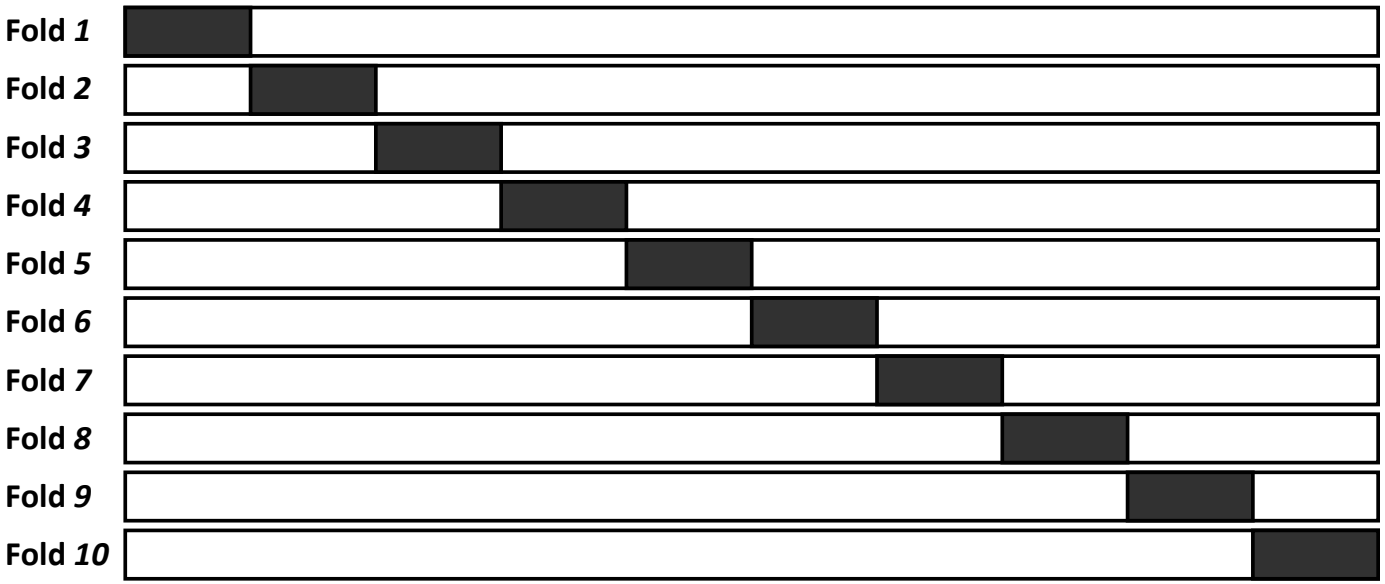
Hold-out sampling can divide the full data into training, validation, and test sets.



Using a validation set to avoid overfitting in iterative machine learning algorithms.

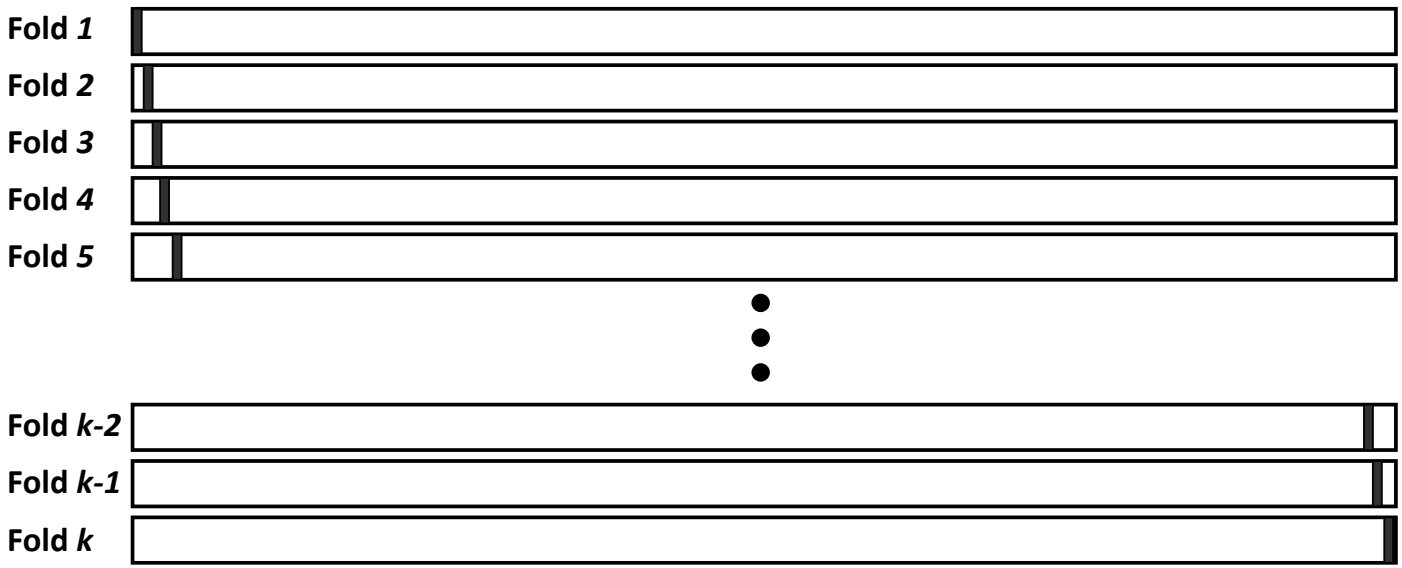
k-Fold Cross Validation

Fold	Confusion Matrix				Class Accuracy
1			Prediction *'lateral'*	Prediction *'frontal'*	81%
	Target	*'lateral'*	43	9	
		'frontal'	10	38	
2			Prediction *'lateral'*	Prediction *'frontal'*	88%
	Target	*'lateral'*	46	9	
		'frontal'	3	42	
3			Prediction *'lateral'*	Prediction *'frontal'*	82%
	Target	*'lateral'*	51	10	
		'frontal'	8	31	
4			Prediction *'lateral'*	Prediction *'frontal'*	85%
	Target	*'lateral'*	51	8	
		'frontal'	7	34	
5			Prediction *'lateral'*	Prediction *'frontal'*	84%
	Target	*'lateral'*	46	9	
		'frontal'	7	38	
			Prediction *'lateral'*	Prediction *'frontal'*	



The division of data during the ***k*-fold cross validation** process. Black rectangles indicate test data, and white spaces indicate training data.

Leave-one-out Cross Validation



The division of data during the **leave-one-out cross validation** process. Black rectangles indicate instances in the test set, and white spaces indicate training data.

Performance Measures: Categorical Targets

Confusion Matrix-based Performance Measures

$$\text{TPR} = \frac{TP}{(TP + FN)} \quad (3)$$

$$\text{TNR} = \frac{TN}{(TN + FP)} \quad (4)$$

$$\text{FPR} = \frac{FP}{(TN + FP)} \quad (5)$$

$$\text{FNR} = \frac{FN}{(TP + FN)} \quad (6)$$

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

$$\text{TNR} = \frac{9}{(9+2)} = 0.818$$

$$\text{FPR} = \frac{2}{(9+2)} = 0.182$$

$$\text{FNR} = \frac{3}{(6+3)} = 0.333$$

Precision, Recall and F_1 Measure

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

$$\text{precision} = \frac{TP}{(TP + FP)} \quad (7)$$

$$\text{recall} = \frac{TP}{(TP + FN)} \quad (8)$$

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

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

$$\text{F}_1\text{-measure} = 2 \times \frac{(\text{precision} \times \text{recall})}{(\text{precision} + \text{recall})} \tag{9}$$

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

Average Class Accuracy

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

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

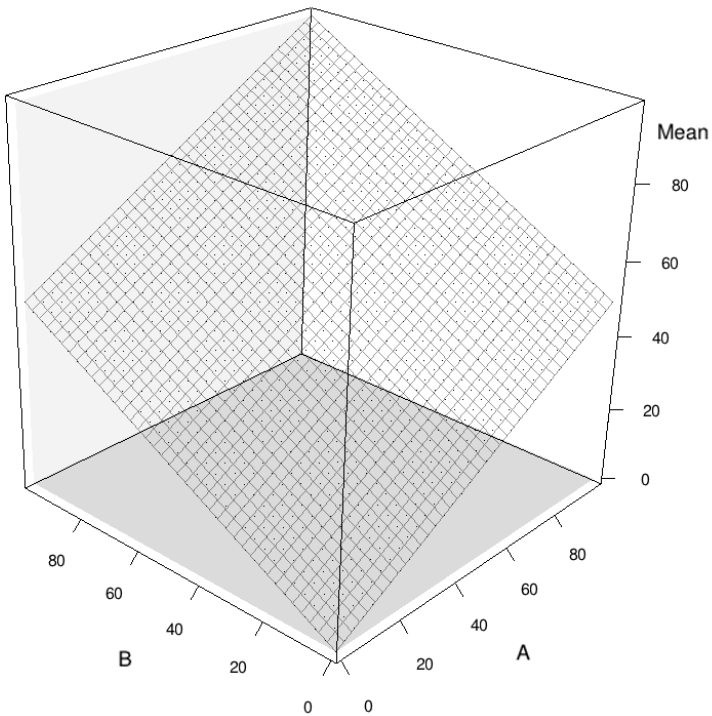
		Prediction	
		'non-churn'	***'churn'***
Target	***'non-churn'***	70	20
	'churn'	2	8

$$\text{average class accuracy} = \frac{1}{|\mathit{levels}(t)|} \sum_{l \in \mathit{levels}(t)} \text{recall}_l \tag{10}$$

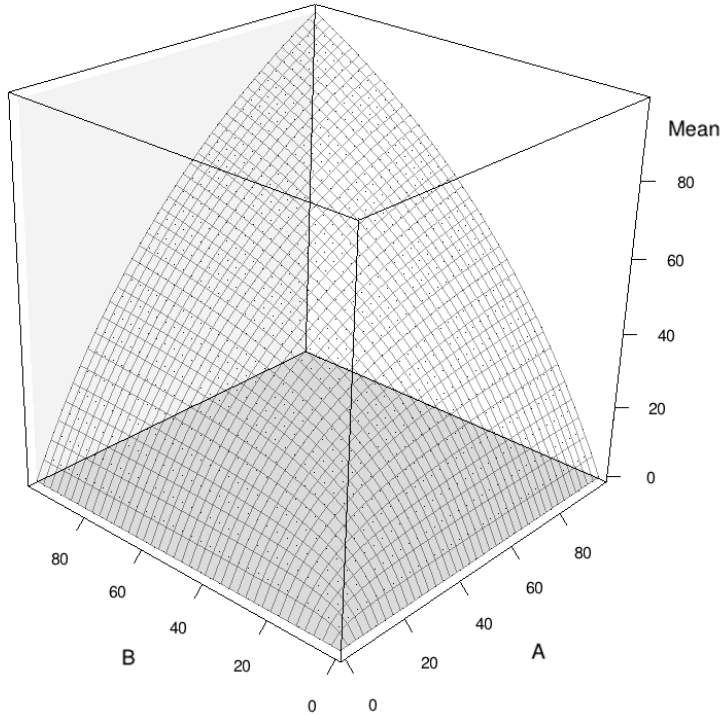
$$\text{average class accuracy}_{\text{HM}} = \frac{1}{\frac{1}{|\mathit{levels}(t)|} \sum_{l \in \mathit{levels}(t)} \frac{1}{\text{recall}_l}} \tag{11}$$

$$\frac{1}{\frac{1}{2}\left(\frac{1}{1.0} + \frac{1}{0.1}\right)} = \frac{1}{5.5} = 18.2\%$$

$$\frac{1}{\frac{1}{2}\left(\frac{1}{0.778} + \frac{1}{0.800}\right)} = \frac{1}{1.268} = 78.873\%$$



(c)



(d)

Surfaces generated by calculating (c) the **arithmetic mean** and (d) the **harmonic mean** of all combinations of features A and B that range from 0 to 100.

Measuring Profit and Loss

- It is not always correct to treat all outcomes equally
- In these cases, it is useful to take into account the cost of the different outcomes when evaluating models

The structure of a **profit matrix**.

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

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

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

Model Performance and Profit Analysis

Confusion matrices for pay-day loan credit scoring: k-NN model vs. decision tree model.

k-NN model ($\text{acc}_{HM} = 83.82\%$)

Target	Good	Bad
Good	57	3
Bad	10	30

Decision tree ($\text{acc}_{HM} = 80.76\%$)

Target	Good	Bad
Good	43	17
Bad	3	37

Overall profit

k-NN model

Target	Good	Bad
Good	7980	-420
Bad	-7000	0
Profit	560	

Decision tree

Target	Good	Bad
Good	6020	-2380
Bad	-2100	0
Profit	1540	

Performance Measures: Prediction Scores

- Prediction models return a thresholded score.

Example

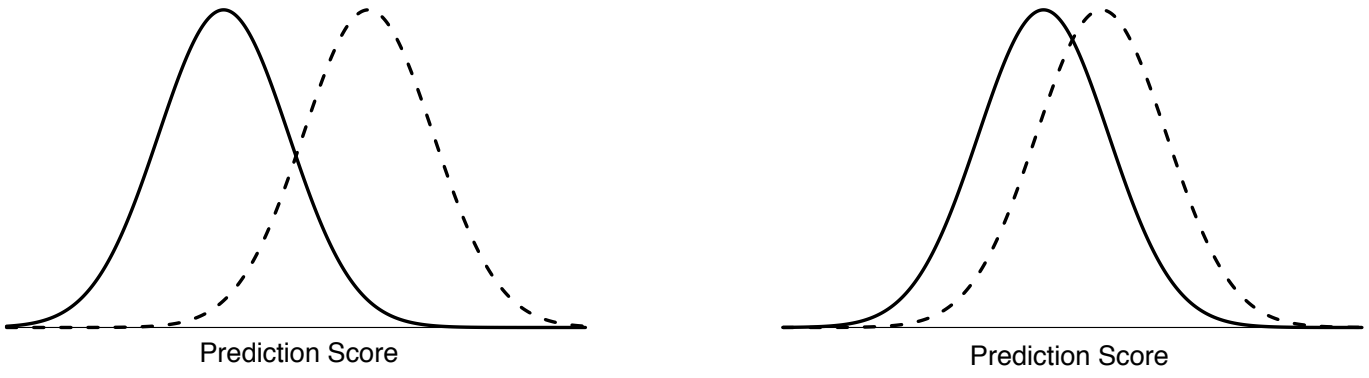
$$threshold(score, 0.5) = \begin{cases} positive & \text{if } score \geq 0.5 \\ negative & otherwise \end{cases} \tag{12}$$

A test set with predictions and scores (threshold$=0.5$).

ID	Target	Prediction	Score	Outcome
7	ham	ham	0.001	TN
11	ham	ham	0.003	TN
15	ham	ham	0.059	TN
13	ham	ham	0.064	TN
19	ham	ham	0.094	TN
12	spam	ham	0.160	FN
2	spam	ham	0.184	FN
3	ham	ham	0.226	TN
16	ham	ham	0.246	TN
1	spam	ham	0.293	FN

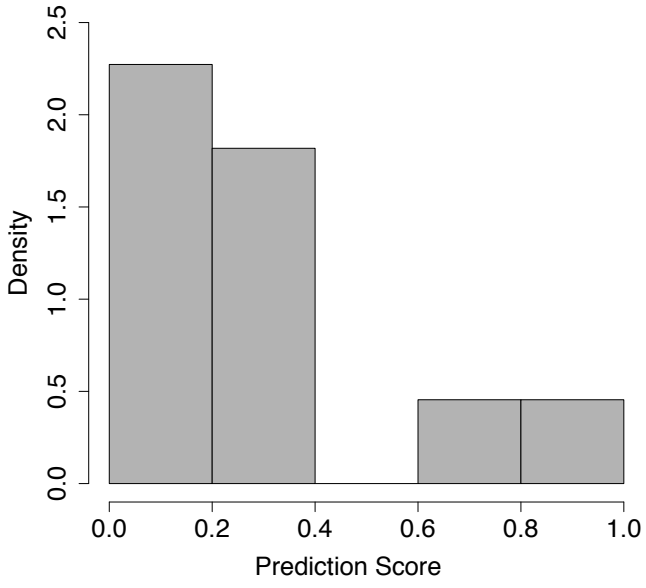
ID	Target	Prediction	Score	Outcome
5	ham	ham	0.302	TN
14	ham	ham	0.348	TN
17	ham	spam	0.657	FP
8	spam	spam	0.676	TP
6	spam	spam	0.719	TP
10	spam	spam	0.781	TP
18	spam	spam	0.833	TP
20	ham	spam	0.877	FP
9	spam	spam	0.960	TP
4	spam	spam	0.963	TP

- Note that instances get a prediction of *'ham'* generally have a low score.
- Some measures use this ability of a model to rank instances that should get predictions of one target level higher than the other, to assess how well the model is performing.
- The basis of most of these approaches is measuring how well the distributions of scores produced by the model for different target levels are separated

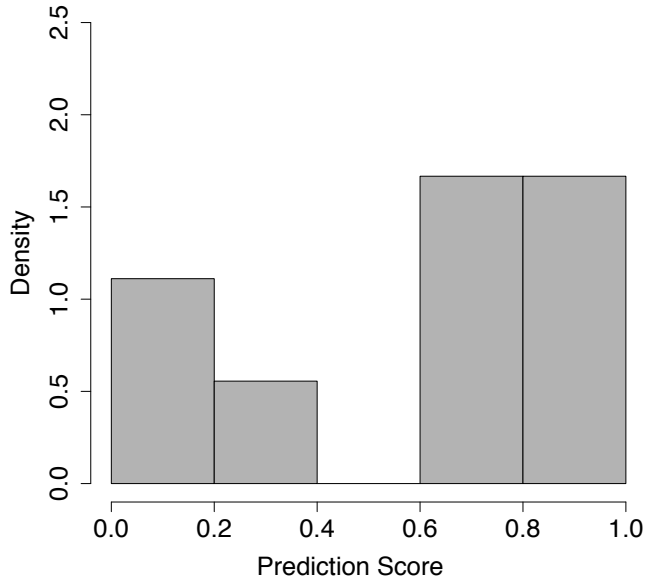


(e) (f)

Prediction score distributions for two different prediction models. The distributions in (e) are much better separated than those in (f).



(g) spam



(h) ham

Prediction score distributions for the *'spam'* and *'ham'* target levels

Receiver Operating Characteristic Curves

- The receiver operating characteristic index (ROC index), which is based on the receiver operating characteristic curve (ROC curve), is a widely used performance measure that is calculated using prediction scores.
- TPR and TNR are intrinsically tied to the threshold used to convert prediction scores into target levels.
- This threshold can be changed, however, which leads to different predictions and a different confusion matrix.

Confusion matrices for the set of predictions using (a) a prediction score threshold of 0.75 and (b) a prediction score threshold of 0.25.

(a) Threshold: 0.75

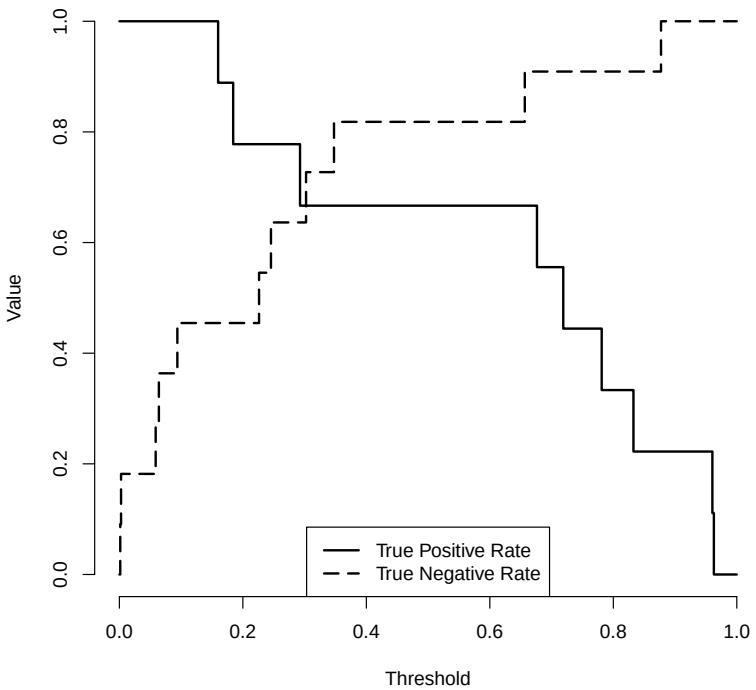
		Prediction	
		'spam'	***'ham'***
Target	***'spam'***	4	4
	'ham'	2	10

(b) Threshold: 0.25

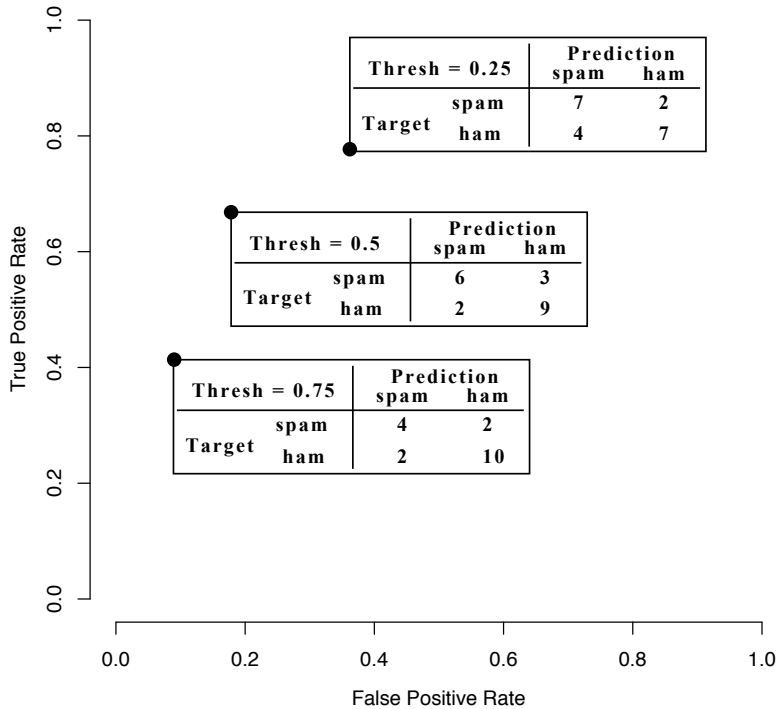
		Prediction	
		'spam'	***'ham'***
Target	***'spam'***	7	2
	'ham'	4	7

ID	Target	Score	Pred. (0.10)	Pred. (0.25)	Pred. (0.50)	Pred. (0.75)	Pred. (0.90)
7	ham	0.001	ham	ham	ham	ham	ham
11	ham	0.003	ham	ham	ham	ham	ham
15	ham	0.059	ham	ham	ham	ham	ham
13	ham	0.064	ham	ham	ham	ham	ham
19	ham	0.094	ham	ham	ham	ham	ham
12	spam	0.160	spam	ham	ham	ham	ham
2	spam	0.184	spam	ham	ham	ham	ham
3	ham	0.226	spam	ham	ham	ham	ham
16	ham	0.246	spam	ham	ham	ham	ham
1	spam	0.293	spam	spam	ham	ham	ham
5	ham	0.302	spam	spam	ham	ham	ham
14	ham	0.348	spam	spam	ham	ham	ham
17	ham	0.657	spam	spam	spam	ham	ham
8	spam	0.676	spam	spam	spam	ham	ham
6	spam	0.719	spam	spam	spam	ham	ham
10	spam	0.781	spam	spam	spam	spam	ham
18	spam	0.833	spam	spam	spam	spam	ham
20	ham	0.877	spam	spam	spam	spam	ham
9	spam	0.960	spam	spam	spam	spam	spam
4	spam	0.963	spam	spam	spam	spam	spam
Misclassification Rate			0.300	0.300	0.250	0.300	0.350
True Positive Rate (TPR)			1.000	0.778	0.667	0.444	0.222
True Negative rate (TNR)			0.455	0.636	0.818	0.909	1.000
False Positive Rate (FPR)			0.545	0.364	0.182	0.091	0.000
False Negative Rate (FNR)			0.000	0.222	0.333	0.556	0.778

- Note: as the threshold increases TPR decreases and TNR increases (and vice versa).
- Capturing this tradeoff is the basis of the ROC curve.

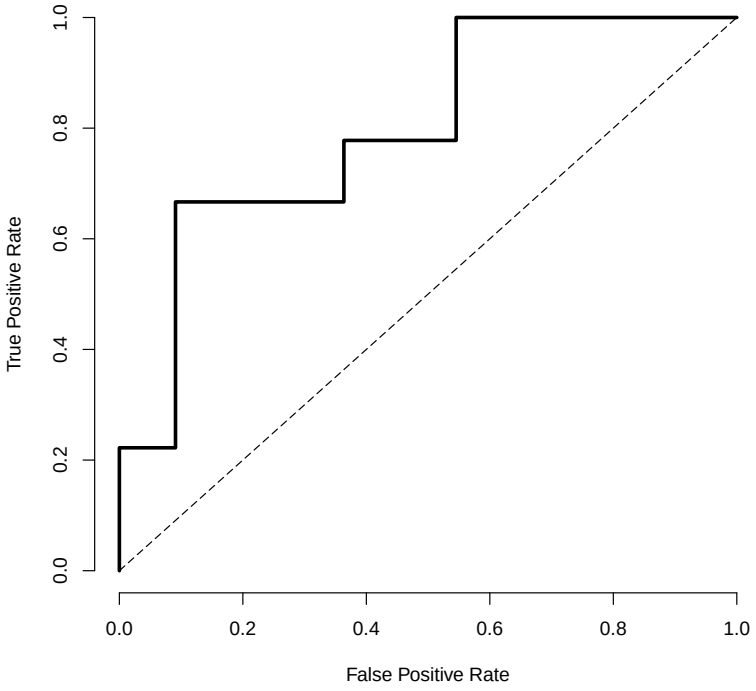


(i)

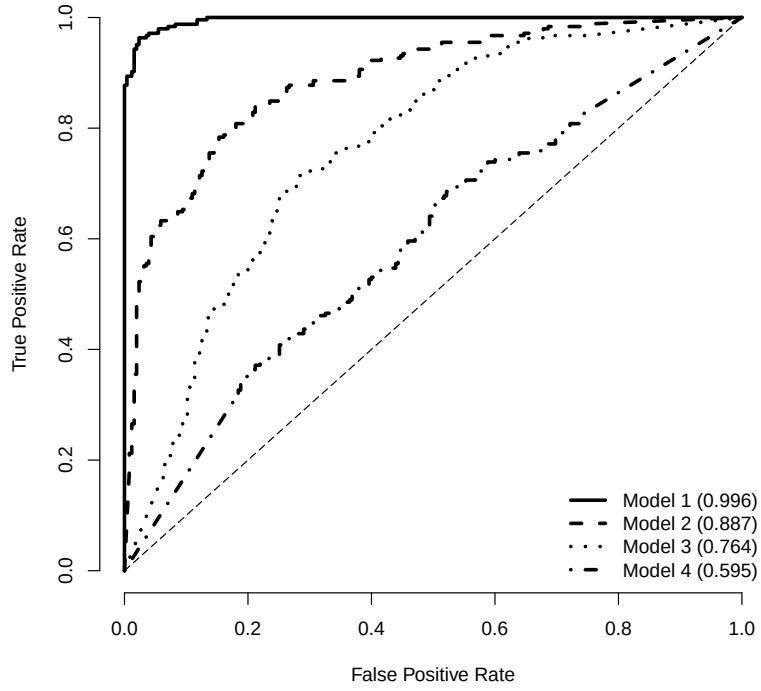


(j)

(i) The changing values of TPR and TNR as the threshold is altered;
(j) points in ROC space for thresholds of 0.25, 0.5, and 0.75.



(k)



(l)

(k) A complete ROC curve for the email classification example; (l) a selection of ROC curves for different models trained on the same prediction task.

Performance Measures: Continuous Targets

Basic Measures of Error

$$\text{sum of squared errors} = \frac{1}{2}\sum_{i=1}^{n}(t_i - \mathbb{M}(\mathbf{d}_i))^2 \tag{13}$$

$$\text{mean squared error} = \frac{\sum_{i=1}^{n}(t_i - \mathbb{M}(\mathbf{d}_i))^2}{n} \tag{14}$$

$$\text{root mean squared error} = \sqrt{\frac{\sum_{i=1}^{n}(t_i - \mathbb{M}(\mathbf{d}_i))^2}{n}} \tag{15}$$

$$\text{mean absolute error} = \frac{\sum_{i=1}^{n} abs(t_i - \mathbb{M}(\mathbf{d}_i))}{n} \tag{16}$$

Domain Independent Measures of Error

$$R^2 = 1 - \frac{\text{sum of squared errors}}{\text{total sum of squares}} \tag{17}$$

$$\text{total sum of squares} = \frac{1}{2}\sum_{i=1}^{n}\left(t_i - \bar{t}\right)^2 \tag{18}$$

		Linear Regression		k-NN	
ID	Target	Prediction	Error	Prediction	Error
1	10.502	10.730	0.228	12.240	1.738
2	18.990	17.578	-1.412	21.000	2.010
3	20.000	21.760	1.760	16.973	-3.027
4	6.883	7.001	0.118	7.543	0.660
5	5.351	5.244	-0.107	8.383	3.032
6	11.120	10.842	-0.278	10.228	-0.892
7	11.420	10.913	-0.507	12.921	1.500
8	4.836	7.401	2.565	7.588	2.752
9	8.177	8.227	0.050	9.277	1.100
10	19.009	16.667	-2.341	21.000	1.991
11	13.282	14.424	1.142	15.496	2.214
12	8.689	9.874	1.185	5.724	-2.965
13	18.050	19.503	1.453	16.449	-1.601
14	5.388	7.020	1.632	6.640	1.252
15	10.646	10.358	-0.288	5.840	-4.805
16	19.612	16.219	-3.393	18.965	-0.646
17	10.576	10.680	0.104	8.941	-1.634
18	12.934	14.337	1.403	12.484	-0.451
19	10.492	10.366	-0.126	13.021	2.529
20	13.439	14.035	0.596	10.920	-2.519
21	9.849	9.821	-0.029	9.920	0.071
22	18.045	16.639	-1.406	18.526	0.482
23	6.413	7.225	0.813	7.719	1.307
24	9.522	9.565	0.043	8.934	-0.588
25	12.083	13.048	0.965	11.241	-0.842
26	10.104	10.085	-0.020	10.010	-0.095
27	8.924	9.048	0.124	8.157	-0.767
28	10.636	10.876	0.239	13.409	2.773
29	5.457	4.080	-1.376	9.684	4.228
30	3.538	7.090	3.551	5.553	2.014
	MSE		**1.905**		**4.394**
	RMSE		**1.380**		**2.096**
	MAE		**0.975**		**1.750**
	R^2		**0.889**		**0.776**

Evaluating Models after Deployment

To monitor model performance, we need a change signal from three possible sources:

1. Model performance metrics
2. Output distributions of the model
3. Feature distributions in the model's input data

Monitoring Changes in Performance Measures

- Continuously monitor the model with the same performance metrics to detect concept drift.
- Compare current performance against pre-deployment results to identify significant changes, signaling potential concept drift.
- Monitoring for performance shifts is straightforward but relies on having immediate access to accurate target values after deployment.

Monitoring Model Output Distributions

- Use changes in the distribution of model outputs as a signal for concept drift.

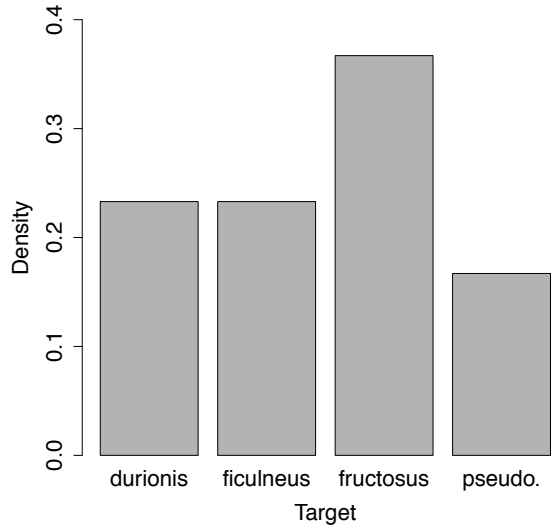
$$\text{stability index} = \sum_{l \in levels(t)} \left(\left(\frac{|\mathcal{A}_{t=l}|}{|\mathcal{A}|} - \frac{|\mathcal{B}_{t=l}|}{|\mathcal{B}|} \right) \times log_e \left(\frac{|\mathcal{A}_{t=l}|}{|\mathcal{A}|} / \frac{|\mathcal{B}_{t=l}|}{|\mathcal{B}|} \right) \right) \quad (19)$$

- stability index < 0.1, the distribution of the newly collected test set is similar to that in the original one.
- stability index [0.1, 0.25], some change has occurred.
- stability index > 0.25, a significant change has occurred.

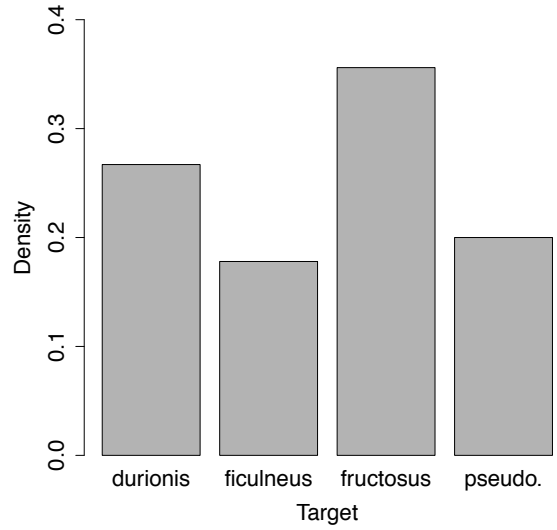
Calculating the **stability index** for the bacterial species identification problem given new test data for two periods after model deployment. The frequency and percentage of each target level are shown for the original test set and for two samples collected after deployment. The column marked SI_t shows the different parts of the stability index sum

	Original		New Sample 1			New Sample 2		
Target	Count	%	Count	%	SI_t	Count	%	SI_t
'durionis'	7	0.233	12	0.267	0.004	12	0.200	0.005
'ficulneus'	7	0.233	8	0.178	0.015	9	0.150	0.037
'fructosus'	11	0.367	16	0.356	0.000	14	0.233	0.060
'pseudo.'	5	0.167	9	0.200	0.006	25	0.417	0.229
Sum	30		45		**0.026**	60		**0.331**

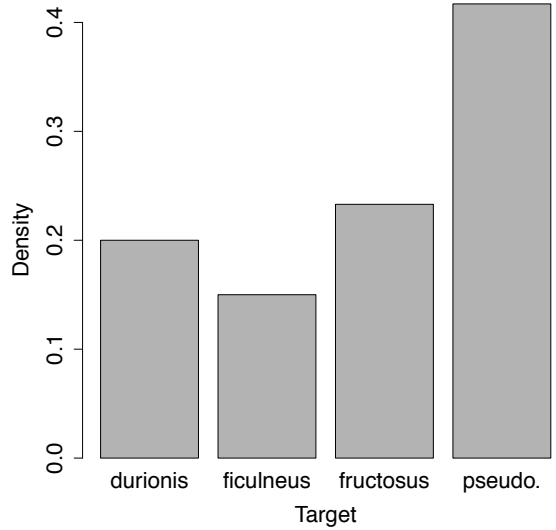
$$
\begin{aligned}
\text{stability index} &= \left(\frac{7}{30} - \frac{12}{45}\right) \times log_e\left(\frac{7}{30} / \frac{12}{45}\right) \\
&\quad + \left(\frac{7}{30} - \frac{8}{45}\right) \times log_e\left(\frac{7}{30} / \frac{8}{45}\right) \\
&\quad + \left(\frac{11}{30} - \frac{16}{45}\right) \times log_e\left(\frac{11}{30} / \frac{16}{45}\right) \\
&\quad + \left(\frac{5}{30} - \frac{9}{45}\right) \times log_e\left(\frac{5}{30} / \frac{9}{45}\right) \\
&= 0.026
\end{aligned}
$$



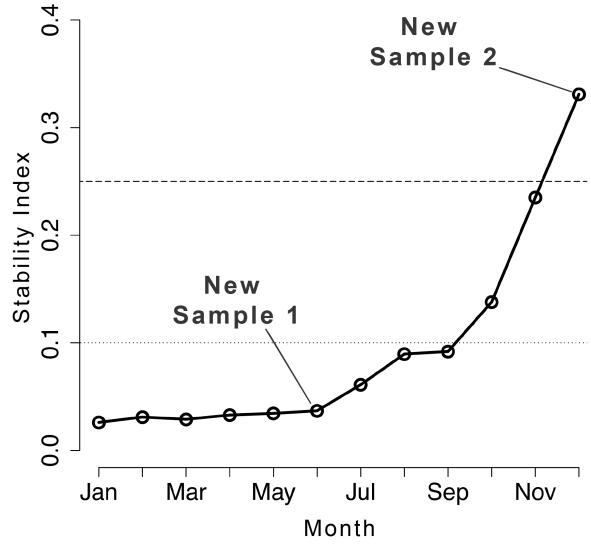
(m) Original



(n) New Sample 1



(o) New Sample 2



(p) Monitoring Over Time

Monitoring Descriptive Feature Distribution Changes

- Compare the distributions of the model's descriptive features.
- Utilize measures like the stability index, χ^2 statistic, or K-S statistic to detect distribution differences.
- Changes in one feature typically do not significantly impact performance in multi-feature models.
- This approach is not recommended for models using many descriptive features (more than 10).

Comparative Experiments Using a Control Group

- Control groups are used not to assess the models' predictive power, but to evaluate their effectiveness in addressing business problems upon deployment.

The number of customers who left the mobile network operator from both the control group (random selection) and the treatment group (model selection).

Week	Control Group (Random Selection)	Treatment Group (Model Selection)
1	21	23
2	18	15
3	28	18
4	19	20
5	18	15
6	17	17
7	23	18
8	24	20
9	19	18
10	20	19
11	18	13
12	21	16
Mean	20.500	17.667
Std. Dev.	3.177	2.708

- Fewer customers churn when the prediction model is used to select which customers to call.