Ollscoil na Gaillimhe
University of Galway

CT4101
Machine Learning

Dr. Frank Glavin
Room CSB-3004, Computer Science Building
Frank.Glavin@University*of*Galway.ie

School of Computer Science

University *of* Galway.ie

Model Evaluation

Summary of Topic

Fundamental ideas
Using a hold-out test set
Confusion matrices for binary classification problems
Cross validation and grid search
ROC curves
Confusion matrices for multinominal classification problems

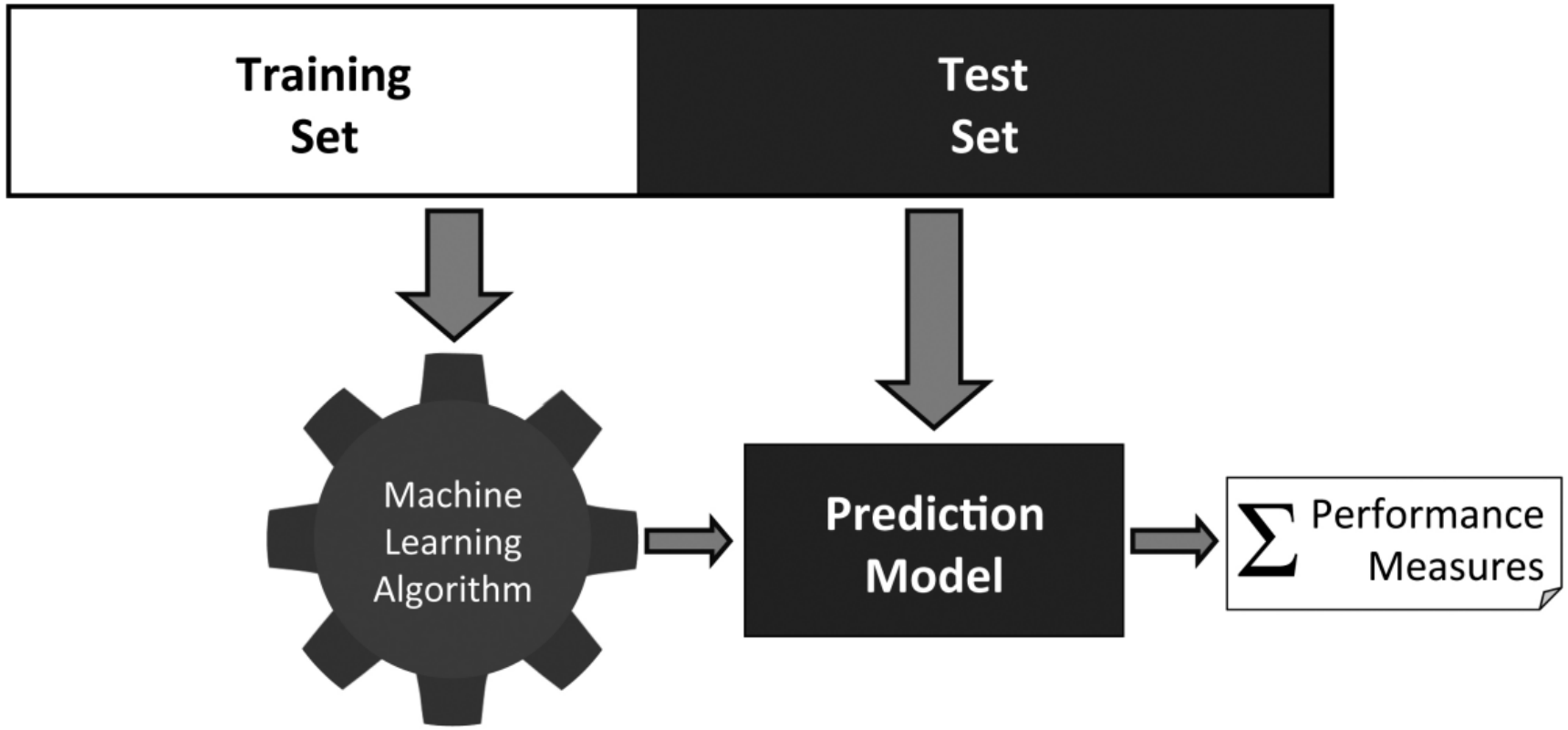
Designing experiments to evaluate ML models

The most important part of the design of an evaluation experiment for a predictive model is ensuring that the data used to evaluate the model is not the same as the data used to train the model.

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 demonstrate to users that the model will meet their needs

Using a hold-out test set



A sample test set

By convention, in binary classification tasks we refer to one class as ‘positive’ and the other class as ‘negative’.
In this example, ‘spam’ is the positive level, and ‘ham’ is the negative level

There are 4 possible outcomes for a binary classification task:

TP = True Positive, TN = True Negative, FP = False Positive, FN = False Negative

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

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

The structure of a confusion matrix for binary classification tasks

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

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Sample confusion matrix (binary classification tasks)

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

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

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Calculating the misclassification rate

$$\text{misclassification rate} = \frac{\text{number incorrect predictions}}{\text{total predictions}} \quad (1)$$

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

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Calculating misclassification rate using a confusion matrix (binary classification tasks)

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

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

$$\text{misclassification accuracy} = \frac{(FP + FN)}{(TP + TN + FP + FN)}$$

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

Calculating accuracy using a confusion matrix (binary classification tasks)

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

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

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

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

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Model Evaluation

Metrics for binary classification tasks

Ollscoil na Gaillimhe
University of Galway

Confusion matrix-based measures for binary categorical targets

TPR = True Positive Rate
TNR = True Negative Rate
FPR = False Positive Rate
FNR = False Negative Rate

All these measures can have values in the range of 0 to 1.
Higher values of TPR and TNR indicate better model performance
Lower values of FNR and FPR indicate better model performance

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

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

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

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

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Calculating TPR, TNR, FPR & FNR

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

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

Calculating these measures using the example from earlier, we can see that the model is much better at predicting 'ham' (TNR) than it is at predicting 'spam' (TPR)

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

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

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

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

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Precision and recall

Precision captures how often, when a model makes a positive prediction, this prediction turns out to be correct.

Recall is equivalent to the true positive rate, and tells us how confident we can be that all the instances with the positive target level have been found by the model

Both precision and recall have values in range 0 to 1

Assuming again in our spam classification example that 'spam' is the positive level:

- precision measures how often the emails marked as 'spam' actually are spam
- recall measures how often the spam messages in the test set were actually marked as 'spam'

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

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

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Calculating precision and recall

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

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

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

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

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

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

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Model Evaluation

Multinominal classification tasks

Ollscoil na Gaillimhe
University of Galway

Calculating performance on multinominal targets

All the measures we discussed to date apply to binary classification problems only
Many classification problems are multinominal (i.e., >2 target levels/classes)
We can easily extend the confusion matrix concept to multiple target levels by adding a row and a column for each level (in this case *l* levels)

		Prediction					Recall
		level1	*level2*	*level3*	⋯	*levell*	
Target	*level1*	-	-	-		-	-
	level2	-	-	-		-	-
	level3	-	-	-		-	-
	⋮				⋱		⋮
	levell	-	-	-		-	-
	Precision	-	-	-	⋯	-	

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Calculating performance on multinominal targets

We can also calculate precision and recall for each target level independently

TP(l) refers the number of instances correctly assigned a prediction of class l
FP(l) refers to the number of instances that are incorrectly assigned a prediction of class l
FN(l) refers to the number of instances that should have been given a prediction of class l but were given some other prediction

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

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

Sample multinominal classification predictions (bacterial species)

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

Ollscoil na Gaillimhe
University of Galway

Sample multinominal confusion matrix

		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	

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

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

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Confusion matrices in scikit-learn

Confusion matrices can easily be created in scikit-learn using the built-in classes
See link below for further information
https://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html

Model evaluation

Cross validation and grid search

Ollscoil na Gaillimhe
University of Galway

Motivation

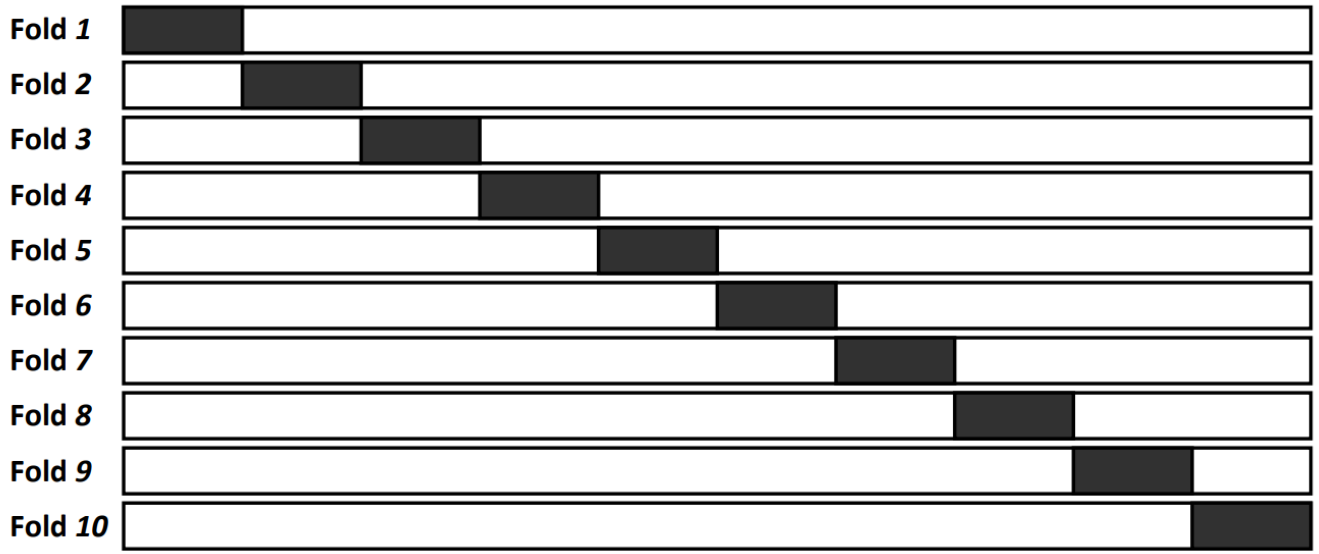
So far we have just manually divided data into training and test sets

Avoid problems with **“lucky split”,** where most difficult examples end up in training set, and most easy examples end up in test set

Methods like *k*-fold cross validation (next slide) allow us to use all examples for both training and testing

k-fold cross validation

k-fold cross validation (CV) allows all of the data to be used for both training and testing
Procedure: split the data into *k* different folds. Use *k*-1 folds for training and the remaining fold for testing. Repeat the entire process *k* times, using a different fold for testing each time. Report per-fold accuracy and average accuracy for both training and testing
The figure shows the process for 10-fold CV (commonly used value). Black rectangles show test data, white spaces show training data



Example results of 5-fold CV

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

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

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

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

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	
Overall			Prediction *'lateral'*	Prediction *'frontal'*	84%
	Target	*'lateral'*	237	45	
		'frontal'	35	183	

TPR: 43/43 + 9 = 0.83
FNR: 9/43 + 9 = 0.17

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Cross validation in scikit-learn (1/2)

Cross validation can easily be implemented in scikit-learn by calling the sklearn.model_selection.cross_val_score() helper function on the estimator and the dataset. This will return testing accuracy only, e.g.

```
>>> from sklearn.model_selection import cross_val_score
>>> clf = svm.SVC(kernel='linear', C=1, random_state=42)
>>> scores = cross_val_score(clf, X, y, cv=5)
>>> scores
array([0.96..., 1. , 0.96..., 0.96..., 1. ])
```

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html
https://scikit-learn.org/stable/modules/cross_validation.html#computing-cross-validated-metrics

Ollscoil na Gaillimhe
University of Galway

School of Computer Science

Cross validation in scikit-learn (2/2)

If it is desired to report training accuracy, or other metrics, scikit-learn provides the function sklearn.model_selection.cross_validate – this function provides additional options

Training scores will be returned if the parameter return_train_score of this function is set to true

The score parameter can be used to specify the metric(s) that will be computed to score the models trained during CV

Hyperparameter Optimisation - Combining Cross Validation with Grid Search

Scikit-learn provides a class sklearn.model_selection.GridSearchCV that allows an exhaustive search through ranges of specified hyperparameter values

Scores are calculated for each possible hyperparameter combination on the grid – this allows the combination with the best score to be identified

Widely used when performing hyperparameter tuning for ML models

See the below links for more details:
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
https://scikit-learn.org/stable/modules/grid_search.html#exhaustive-grid-search

Model evaluation

Prediction scores & ROC curves

Ollscoil na Gaillimhe
University of Galway

Performance measures – prediction scores

Many different classification algorithms produce **prediction scores** (e.g., naïve Bayes, logistic regression, % of positive examples at leaf node of a decision tree)

The score indicates the system's certainty that the given observation belongs to the positive class
To make the decision about whether the observation should be classified as positive or negative, as a consumer of this score, you will interpret the score by picking a classification threshold (cut-off) and compare the score against it

Any observations with scores higher than the threshold are then predicted as the positive class and scores lower than the threshold are predicted as the negative class.
Often, by default a prediction score >= 0.5 is interpreted as the positive class, with values <0.5 interpreted as the negative class

The prediction score threshold can be changed, leading to different performance metric values and a different confusion matrix

Table: A sample test set with model 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

The effect of varying the prediction score threshold

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

Ollscoil na Gaillimhe
University of Galway

Changing the threshold can give different confusion matrices

(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

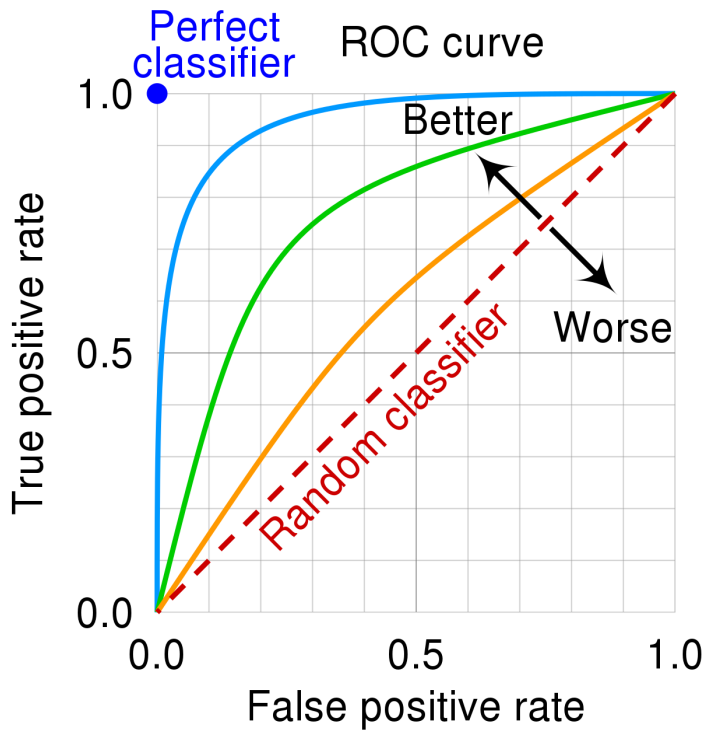
Ollscoil na Gaillimhe
University of Galway

School of Computer Science

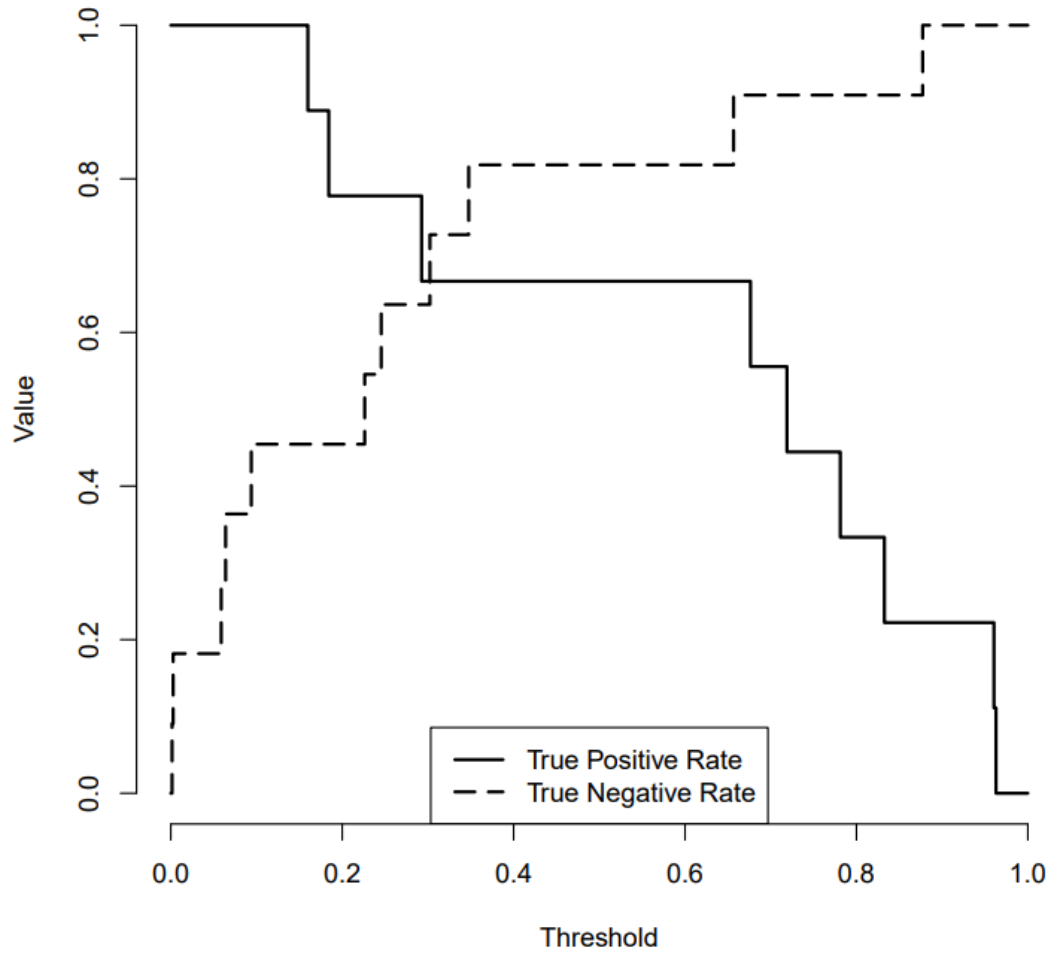
Effect of increasing the threshold

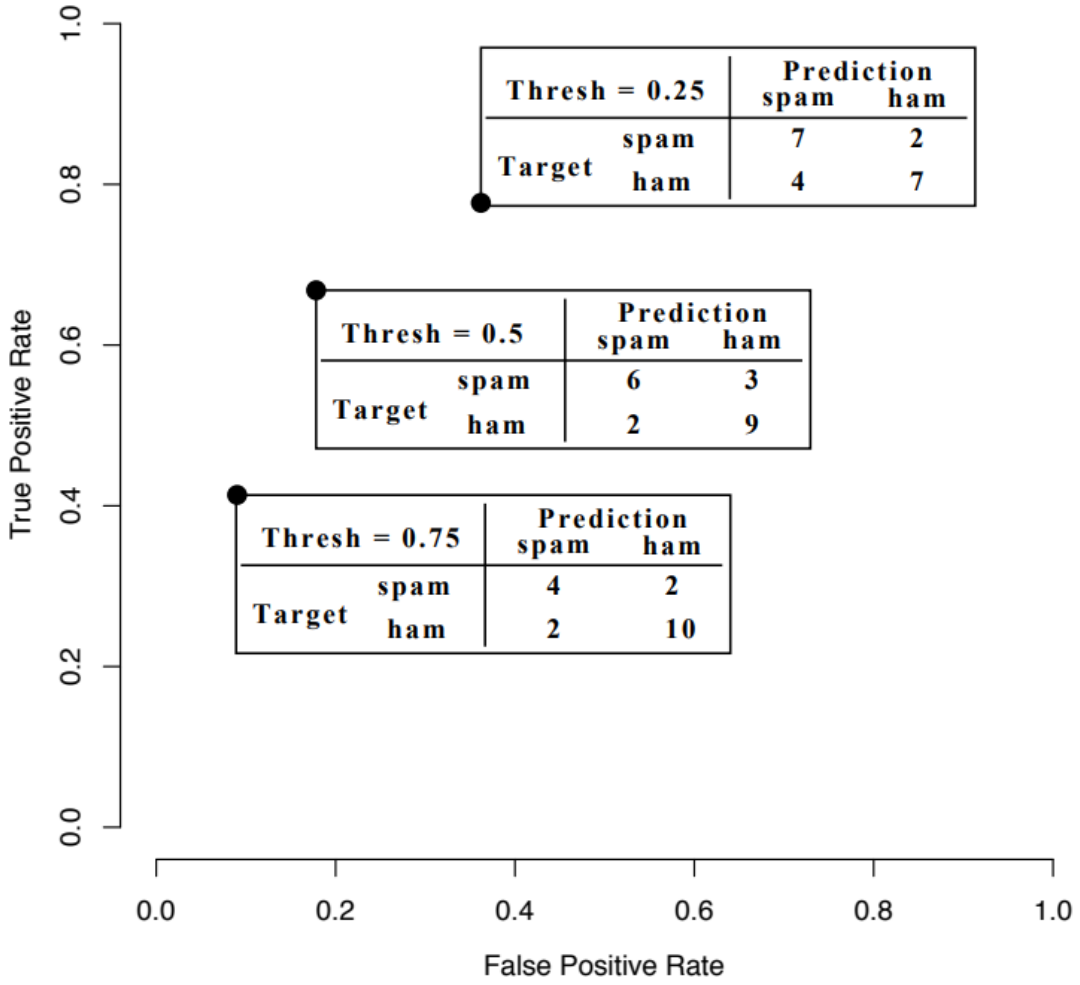
As the threshold increases TPR decreases and TNR increases (and vice versa)

Capturing this trade-off is the basis of the **R**eceiver **O**perating **C**haracteristic curve



Effect of increasing the threshold





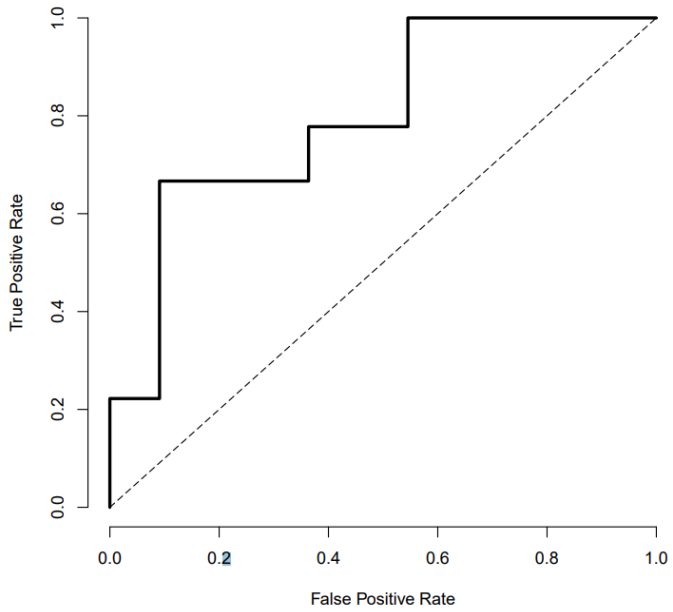
Thresh = 0.25		Prediction spam	Prediction ham
Target	spam	7	2
	ham	4	7

Thresh = 0.5		Prediction spam	Prediction ham
Target	spam	6	3
	ham	2	9

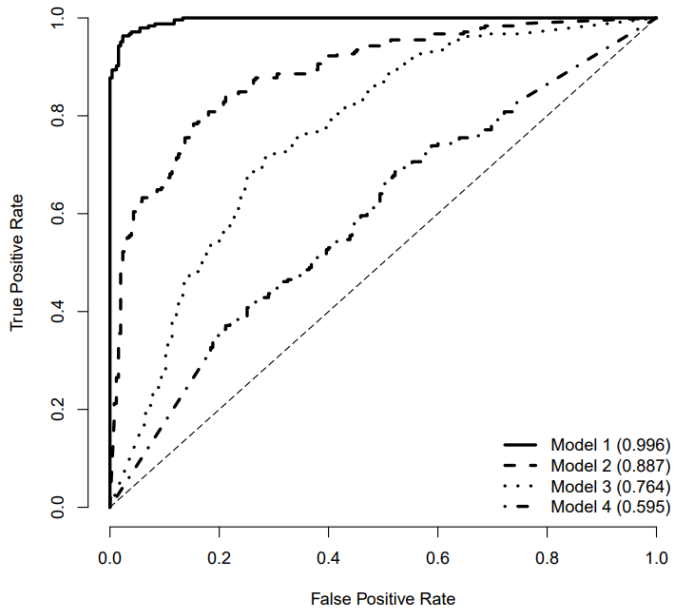
Thresh = 0.75		Prediction spam	Prediction ham
Target	spam	4	2
	ham	2	10

ROC curve examples

An ideal classifier would be in the top left corner (i.e., TPR = 1.0, FPR = 0.0)
The area under the ROC curve often used to determine “strength” of a classifier (greater area means a better classifier, an ideal classifier has an area of 1.0)



ROC curve incorporating the data in the table on slide 26



Sample ROC curves for 5 different models trained on this dataset

ROC curves in scikit-learn

Scikit-learn provides various classes to generate ROC curves
See the link below for more info:
https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html
This link also gives some details on how ROC curves can be computed for multi-class classification problems
(all previous examples in the slides were for binary classification problems)