Evaluation

Sarah Jane Delany

Adapted from ML for DPA book

Introduction

- Evaluation is a critical task in machine learning
- Evaluation is used to
  - To estimate how a model will perform when deployed
  - To compare the performance of different models on a particular task
- Evaluation experiments generally follow these steps:
  - Decide on how to measure performance (ie. evaluation measures to use)
  - Run the model one or more times on a dataset or a collection of datasets
  - Compare the performance of the model with existing benchmark models using the evaluation measure(s)
- The exact experimental setup depends on the hypothesis that is being tested

Hypothesis Testing

- Goal of hypothesis testing: formally examine two opposing hypotheses H_0 and H_A. These two hypotheses are mutually exclusive, so one is true to the exclusion of the other.

- Null Hypothesis H_0: States the assumption to be tested.
  e.g. There is no difference between the performance of two machine learning algorithms.

- *Type I error*: Rejecting H_0 when it is in fact true.
  This is a “false alarm” or "false positive" - detecting a difference, when none actually exists.

- *Type II error*: Failing to reject H_0 when it is in fact false.
  This is a "false negative" - concluding there is no difference, when there really is a difference.

Hypothesis testing

- In ML we often talk about hypotheses at different levels,

- High-level
  - e.g. “*A Neural Network will perform better than Naive Bayes on this particular problem*”

- Low-level   (the output from a model)
  - e.g. “*This email is spam*”

Type I and Type II errors

- *Type I error*: Rejecting H_0 when it is in fact true.
  i.e. “false positive” - detecting a difference, when none exists.
- *Type II error*: Failing to reject H_0 when it is in fact false.
  i.e. "false negative" - concluding there is no difference, when there is.

Statistical Test Result

Real World		H_0 Rejected	H_0 Not Rejected
	There is a real difference	**Correct** A Hit	**Type II Error** Missed a real difference
	There is in fact no difference	**Type I Error** False alarm	**Correct** Right to be sceptical of H_A

Type I and Type II errors

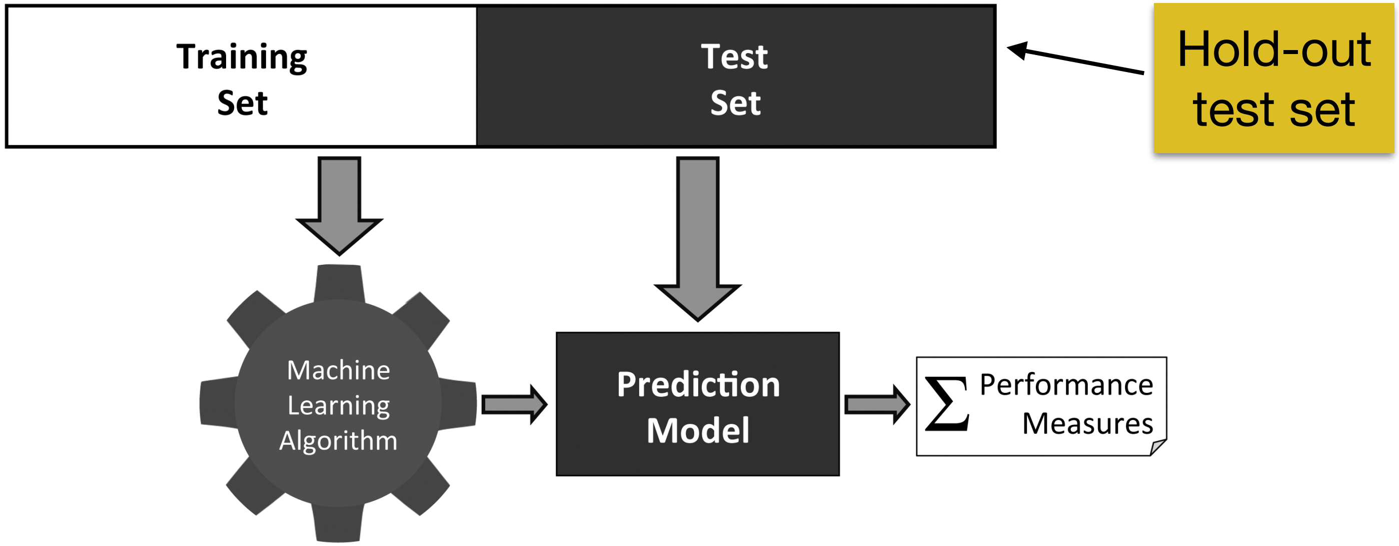
Depending on the hypothesis being tested, these errors will have different costs associated with them.

Null Hypothesis	**Type I Error** **"False Positive"**	**Type II Error** **"False Negative"**
“Person X does not have the virus”	X is clear but marked as infected.	X is infected but marked as well.
Cost of the error	X is required to isolate for 14 days.	X can infect others.

Null Hypothesis	**Type I Error** **"False Positive"**	**Type II Error** **"False Negative"**
"*Person X is not guilty of the crime*"	X is found guilty of a crime, when they are actually innocent.	X is found not-guilty of a crime, when they did actually commit the crime.
Cost of the error	Social cost of sending an innocent person to prison.	Risk of letting a guilty criminal go free.

Standard Approach to Evaluation

- Use labelled test set of known predictions and compare actual label with prediction to measure performance



Training Set

Test Set

Hold-out test set

Machine Learning Algorithm

Prediction Model

∑ Performance Measures

- Mandatory requirement is that the test data must be separate from the training data
  - Use of same training data for testing can produce unrealistic accuracy results that are “too good to be true”
- Performance on test data estimates how the model can generalise beyond the data used to train it

Measuring performance

- No model will ever be perfect
- Important to consider the performance required by the task
  - Medical diagnosis —> need highly accurate diagnosis
  - Predicting customer response to an ad —> anything better than random will deliver profit
- Must align the evaluation measures with the modelling task

Classification Accuracy

- Simplest measure of classification is Misclassification Rate

$$\text{MR} = \frac{\#\text{incorrect predictions}}{\#\text{predictions}}$$

Test Set

ID	Target	Pred.
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

- Associated measure is Accuracy

$$\text{ACC} = \frac{\#\text{correct predictions}}{\#\text{predictions}}$$

Classification Accuracy

- Simplest measure of classification is Misclassification Rate

$$\text{MR} = \frac{\#\text{incorrect predictions}}{\#\text{predictions}}$$

- Associated measure is Accuracy

$$\text{ACC} = \frac{\#\text{correct predictions}}{\#\text{predictions}}$$

Test Set

ID	Target	Pred.
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

$$\text{MR} = \frac{2}{10} = .2$$

$$\text{ACC} = \frac{8}{10} = .8$$

Confusion Matrix

Confusion matrix summarises the performance of a classifier, when compared with the actual target classes (“ground truth”).

		Predicted Class	
		Pos	**Neg**
Actual Class	**Pos**	**TP** **True Positive** Correct!	**FN** **False Negative** (Type II error)
	Neg	**FP** **False Positive** (Type I error)	**TN** **True Negative** Correct!

Clinical Example: Predict a case as *positive* (person has the disease) or *negative* (person does not have the disease)

- **TP** = Sick people correctly predicted as sick
- **FP** = Healthy people incorrectly predicted as sick
- **TN** = Healthy people correctly predicted as healthy
- **FN** = Sick people incorrectly predicted as healthy

Confusion Matrix

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
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

		Prediction	
		spam	**ham**
Actual	**spam**	6	3
	ham	2	9

Classification Evaluation Measures

- Accuracy

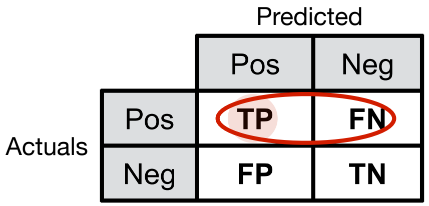
$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

- Misclassification Rate

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

- True Positive Rate:
  Focus on TPs
  aka *Sensitivity &*
  Positive Recall

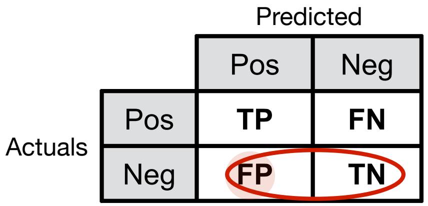
$$\text{TPRate} = \frac{TP}{TP + FN}$$



		Predicted	
		Pos	Neg
Actuals	Pos	TP	FN
	Neg	FP	TN

- False Positive Rate:
  Focus on FPs

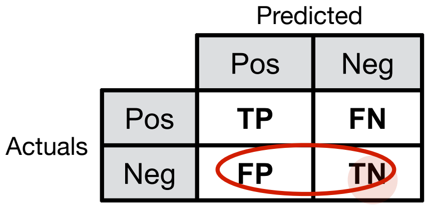
$$\text{FPRate} = \frac{FP}{FP + TN}$$



		Predicted	
		Pos	Neg
Actuals	Pos	TP	FN
	Neg	FP	TN

- True Negative Rate:
  Focus on TNs
  Also called *Specificity*
  & Negative Recall

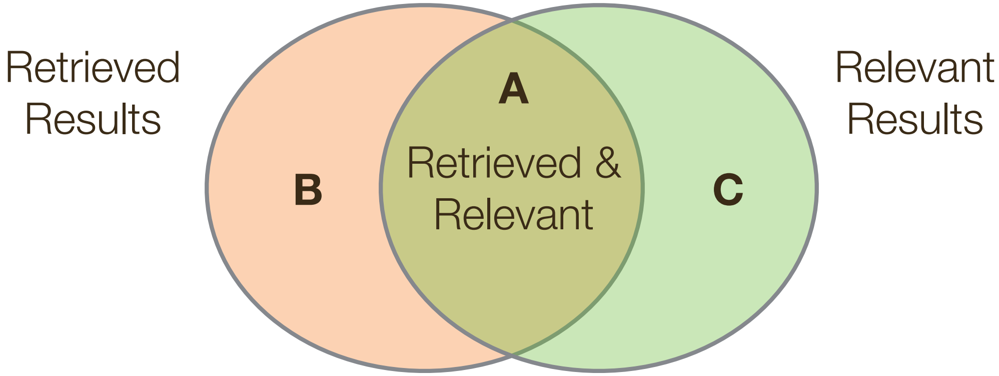
$$\text{TNRate} = \frac{TN}{FP + TN}$$



		Predicted	
		Pos	Neg
Actuals	Pos	TP	FN
	Neg	FP	TN

Precision & Recall

- These are measures that come originally from the Information Retrieval domain
- Precision: proportion of retrieved results that are relevant.
- Recall: proportion of relevant results that are retrieved.



$$\text{Precision} = \frac{A}{A+B}$$

$$\text{Recall} = \frac{A}{A+C}$$

Search Example: Given a collection of 100k documents, we want to find all documents on “hospital waiting lists”. In fact, 45 relevant documents actually exist.

Perform a search, 10 docs were returned, 9 of which are relevant documents.

➡ Precision = 9/10 = 90% of retrieved results were relevant.

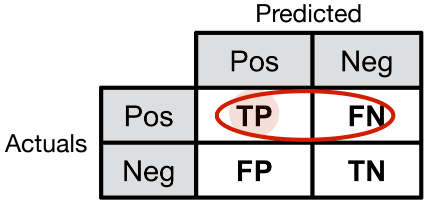
➡ Recall = 9/45 = 20% of all possible relevant results were retrieved

➡ P@10 used in Web Search, how many relevant webpages were returned in top 10

Precision & Recall

- Applying these to ML, these are *class* measures

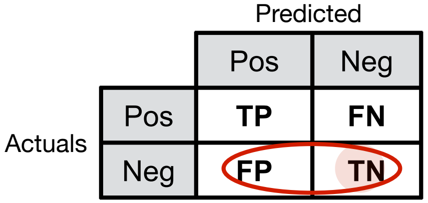
- Recall is class accuracy, see TPRate and TNRate



Predicted

		Pos	Neg
Actuals	Pos	TP	FN
	Neg	FP	TN

Recall



Predicted

		Pos	Neg
Actuals	Pos	TP	FN
	Neg	FP	TN

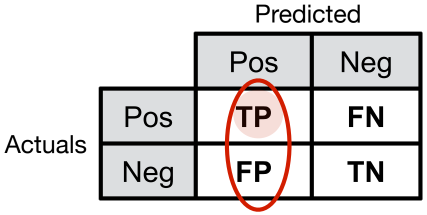
Negative Recall

- Recall is the proportion of the actual instances of the class that are correctly predicted

- Precision - the proportion of the predicted instances of the class that are correct

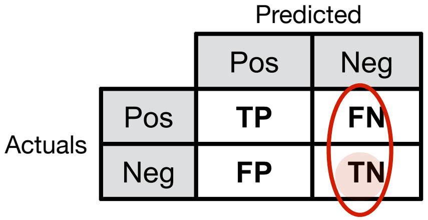
$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Negative Precision} = \frac{TN}{TN + FN}$$



Predicted

		Pos	Neg
Actuals	Pos	TP	FN
	Neg	FP	TN

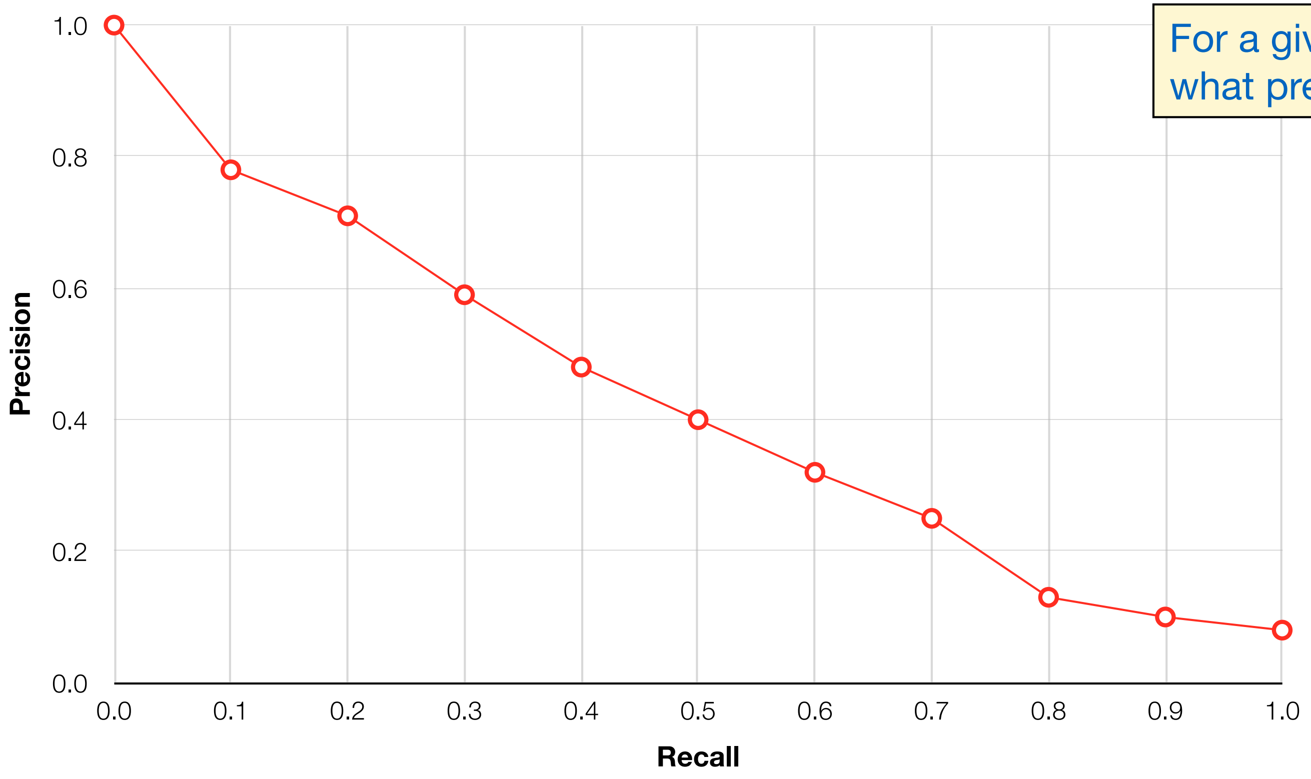


Predicted

		Pos	Neg
Actuals	Pos	TP	FN
	Neg	FP	TN

P/R Curve

- Plot the trade-off between the two measures using a Precision-Recall (PR) curve.
- Used to study the output of a binary classifier.
- Measure precision at fixed recall intervals.
- Improving one with dis-improve the other



For a given level of recall, what precision do we achieve?

Example Calculations

- Accuracy
- Misclassification Rate
- True Positive Rate
- False Positive Rate
- True Negative Rate
- Precision
- Recall

	Label	Prediction	Outcome
1	spam	non-spam	
2	spam	spam	
3	non-spam	non-spam	
4	spam	spam	
5	non-spam	spam	
6	non-spam	non-spam	
7	spam	spam	
8	non-spam	spam	
9	non-spam	non-spam	
10	spam	spam	

Predicted Class

Spam	Non	
?	**?**	Spam
?	**?**	Non

Actual Class

Example Calculations

	Label	**Prediction**	**Outcome**
1	spam	non-spam	FN
2	spam	spam	TP
3	non-spam	non-spam	TN
4	spam	spam	TP
5	non-spam	spam	FP
6	non-spam	non-spam	TN
7	spam	spam	TP
8	non-spam	spam	FP
9	non-spam	non-spam	TN
10	spam	spam	TP

Predicted Class

Spam	Non		
TP=4	**FN=1**	Spam	Actual Class
FP=2	**TN=3**	Non	

F1-Measure

- Precision and Recall can be collapsed to a single performance measure
- F1-Measure, aka F1-score, is the harmonic mean of precision and recall

$$\text{F}_1\text{-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Example Calculations

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

$$= \frac{4+3}{10} = 0.7$$

$$\text{TPRate} = \frac{TP}{TP + FN} = \frac{4}{4+1} = 0.8$$

$$\text{FPRate} = \frac{FP}{FP + TN} = \frac{2}{2+3} = 0.4$$

$$\text{TNRate} = \frac{TN}{FP + TN} = \frac{3}{2+3} = 0.6$$

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{4}{4+2} = 0.667$$

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{4}{4+1} = 0.8$$

	Label	Prediction	Outcome
1	spam	non-spam	FN
2	spam	spam	TP
3	non-spam	non-spam	TN
4	spam	spam	TP
5	non-spam	spam	FP
6	non-spam	non-spam	TN
7	spam	spam	TP
8	non-spam	spam	FP
9	non-spam	non-spam	TN
10	spam	spam	TP

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$F1 = \frac{2 \times 0.667 \times 0.8}{0.667 + 0.8} = 0.727$$

Multiclass Performance

- Macro F1 score - arithmetic mean of all per-class FI scores
- Micro F1 score - global F1 score across all classes

Imbalanced Data

- Imbalanced data: Refers to a problem in classification where the classes in the data are not represented equally (i.e. the distribution of class sizes is skewed).
- **Example:** In a binary classification problem, we have a dataset of 100 items. 80 items belong to Class A, 20 belong to Class B. This is an imbalanced dataset, where the ratio A:B is 4:1. We call A the majority class and B the minority class.
- This phenomenon occurs in many real-world problems:
  - Fraud detection: Vast majority of financial transactions are legitimate, a small minority are fraudulent.
  - Churn analysis: Vast majority of customers stay with their mobile operator, a small minority cancel their subscription.
  - Other examples: medical diagnosis, e-commerce, security.

Example

- Consider the following example:
  In a binary classification problem, a dataset of 100 items. 90 items belong to Class A, 10 belong to Class B.
- Accuracy = ??

Predicted Class

A	B	
90	**0**	A
9	**1**	B

Actual Class

Example

- Consider the following example:
  In a binary classification problem, a dataset of 100 items. 90 items belong to Class A, 10 belong to Class B.
- Accuracy = 91% which seems good!
- The majority class overwhelms the performance of the model on the minority class
- In imbalanced datasets high accuracy can be achieved by just predicting the majority class

Predicted Class

A	B	
90	**0**	A
9	**1**	B

Actual Class

Balanced Measures

- To deal with skewed classes, use a balanced evaluation measure. Measures include:
  - Balanced Accuracy Rate (BAR): Mean of TPRate and TNRate aka Average Class Accuracy (ACA)
  - Balanced Error Rate (BER): Mean of FPRate and FNRate
- For multi-class problems:

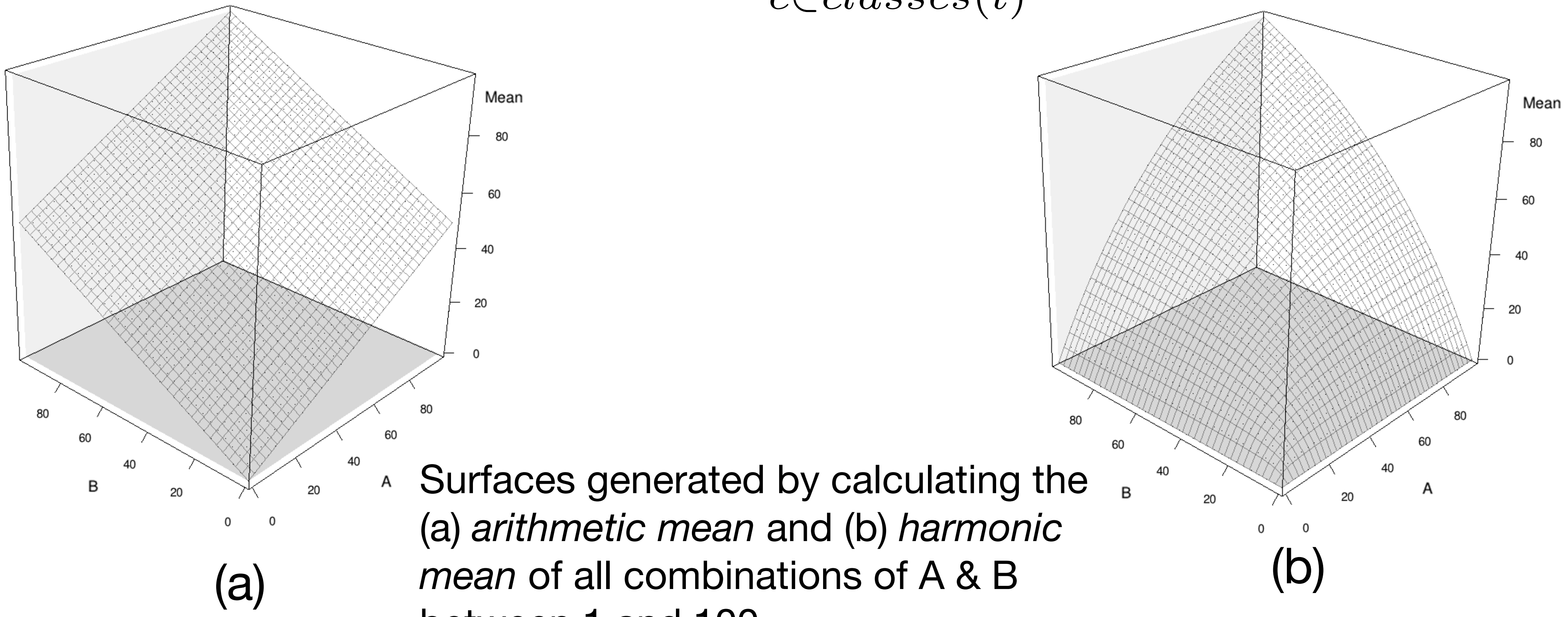
$$\text{ACA} = \frac{1}{|classes(t)|} \sum_{c \in classes(t)} \text{recall}_c$$

- ACA is an *arithmetic mean* which is susceptible to the influence of large outliers

Balanced Measures

- Harmonic Mean mitigates the impact of large outliers and emphasises the importance of smaller values

$$\mathrm{ACA}_{\mathrm{HM}} = \frac{1}{\frac{1}{|classes(t)|} \sum_{c \in classes(t)} \frac{1}{\mathrm{recall}_c}}$$



(a)

(b)

Surfaces generated by calculating the (a) *arithmetic mean* and (b) *harmonic mean* of all combinations of A & B between 1 and 100

F1 Measure is a harmonic mean

Example

- Consider the following example:
  In a binary classification problem, a dataset of 100 items. 90 items belong to Class A, 10 belong to Class B.

Raw Accuracy = 90%

ACA = 55%

ACA$_{HM}$ = 18%

Predicted Class

A	B	Actual Class
90	**0**	A
9	**1**	B

Prediction Score

- Binary Classification models generally do not return a target class as the output, they return a prediction score that is converted to a class label using a threshold
  - NB returns a value for each class and we choose the largest
  - DTrees returns the proportion of each class at a leaf node and we choose the largest
  - kNN returns the weighted distance of the nearest neighbours of each class and we choose the largest
- The prediction score is generally normalised in the range [0,1] and a threshold of 0.5 is used to convert this score to a class

If score is greater than 0.5,
assign to positive class
else assign to negative class

This is effectively the *majority / largest* bit

Previous Example for Naive Bayes:

ID	HEADACHE	FEVER	VOMITING	MENINGITIS
1	true	true	false	false
2	false	true	false	false
3	true	false	true	false
4	true	false	true	false
5	false	true	false	true
6	true	false	true	false
7	true	false	true	false
8	true	false	true	true
9	false	true	false	false
10	true	false	true	true

What is the diagnosis for someone with a headache, fever but no vomiting?

$$\underset{c \in classes(t)}{\arg\max} \prod_{i=1}^{n} P(q_i \mid t = c) \times P(t = c)$$

Two classes: $m=T$ & $m=F$

$m=T$: $P(h=T \mid m=T)$ x $P(f=T \mid m=T)$ x $P(v=F \mid m=T)$ x $P(m=T)$

= 2/3 x 1/3 x 1/3 x 3/10 = 0.0148

$m=F$: $P(h=T \mid m=F)$ x $P(f=T \mid m=F)$ x $P(v=F \mid m=F)$ x $P(m=F)$

= 5/7 x 3/7 x 3/7 x 7/10 = 0.0918

$$\text{score}(m = T) = \frac{0.0148}{0.0148 + 0.0918} = 0.14 \qquad \text{score}(m = F) = \frac{0.0918}{0.0148 + 0.0918} = 0.86$$

Score Threshold

- In some applications the cost of a false positive or false negative is very high, e.g. clinical diagnosis, fraud detection
- The threshold is used to discriminate when selecting between a positive and negative outcome
- The threshold used on the prediction score can be moved to change the sensitivity of the model
- Varying the decision threshold value can lead to different results, and so to different confusion matrices

ROC Analysis

ID	Target	Pred-iction	Score	Out-come
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
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

Threshold 0.5

Predicted Class

Ham	Spam	
TN=9	**FP=2**	Ham
FN=3	**TP=6**	Spam

Actual Class

Threshold 0.75

Predicted Class

Ham	Spam	
TN=10	**FP=1**	Ham
FN=5	**TP=4**	Spam

Actual Class

Threshold 0.25

Predicted Class

Ham	Spam	
TN=7	**FP=4**	Ham
FN=2	**TP=7**	Spam

Actual Class

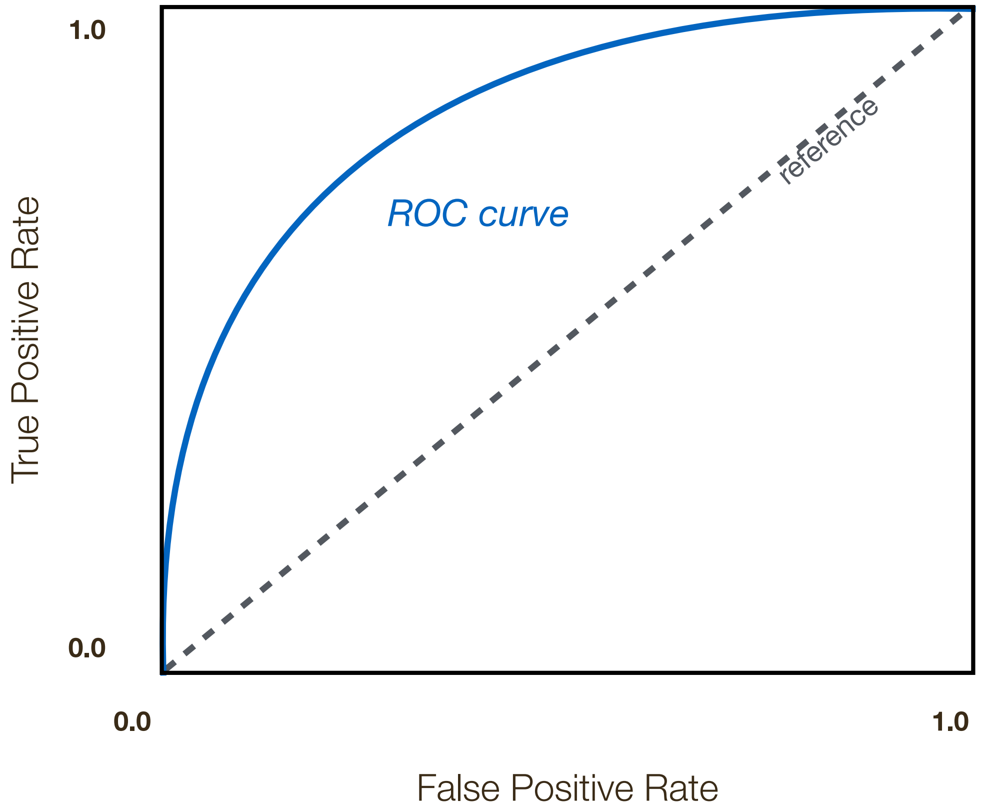
- As threshold increase, TPR decreases, TNR increases although misclassification rate doesn’t change too much

What’s the impact on changing the threshold for this task?

ROC Analysis

- TPR and TNR are intrinsically tied to the threshold used for the prediction score
- There is a trade off between TPR and TNR - the Receiver Operating Characteristic (ROC) Curve captures this trade off

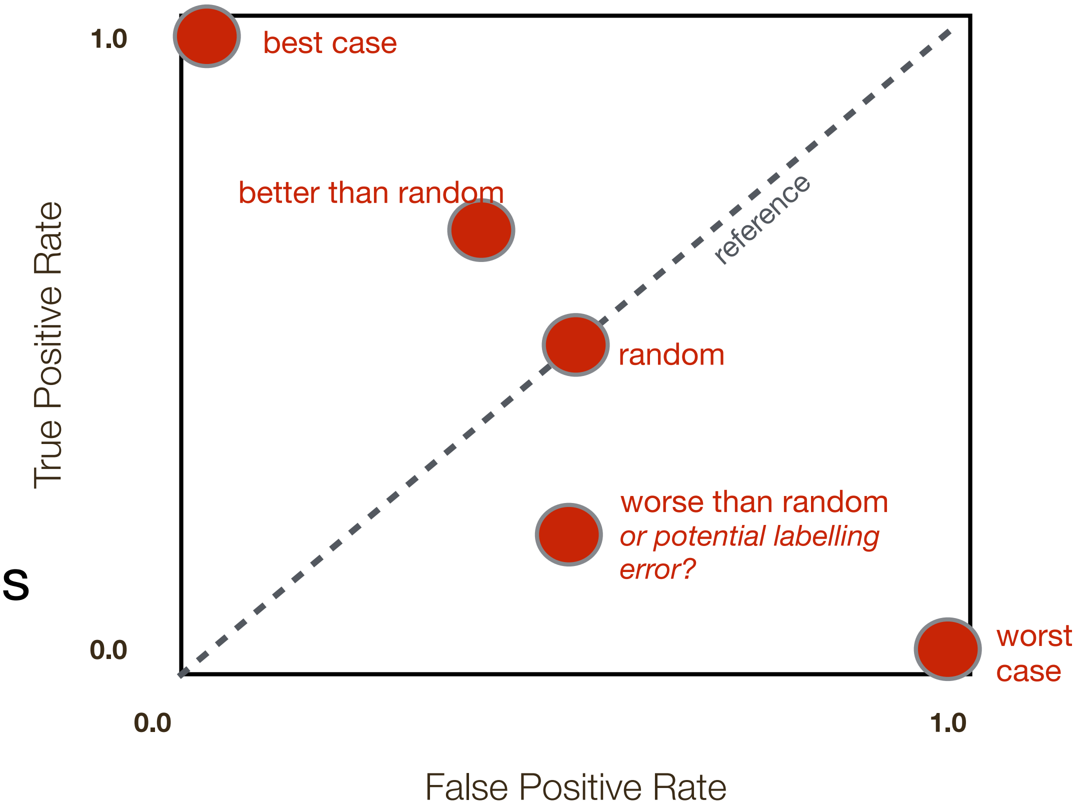
- Plot TPR on vertical axis against 1-TNR (= FPR) on the horizontal axis for a range of score threshold values
- A trained classifier should always be above the 'random' reference line



ROC Analysis

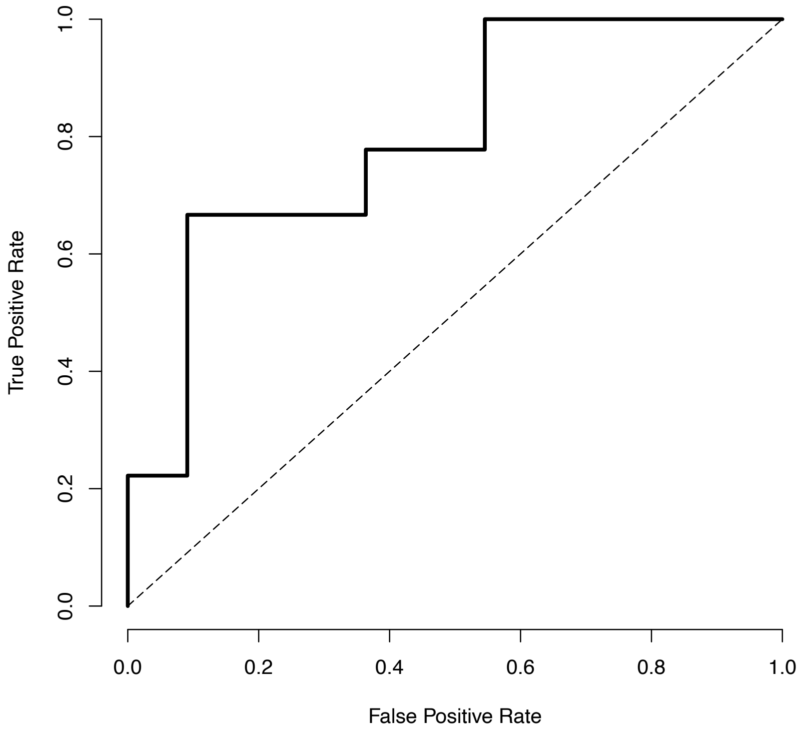
- TPR and TNR are intrinsically tied to the threshold used for the prediction score
- There is a trade off between TPR and TNR - the Receiver Operating Characteristic (ROC) Curve captures this trade off

- Plot TPR on vertical axis against 1-TNR (= FPR) on the horizontal axis for a range of score threshold values
- A trained classifier should always be above the ‘random’ reference line
- As strength of the model increases the ROC moves to top left hand corner (TPR =1; TNR=1)

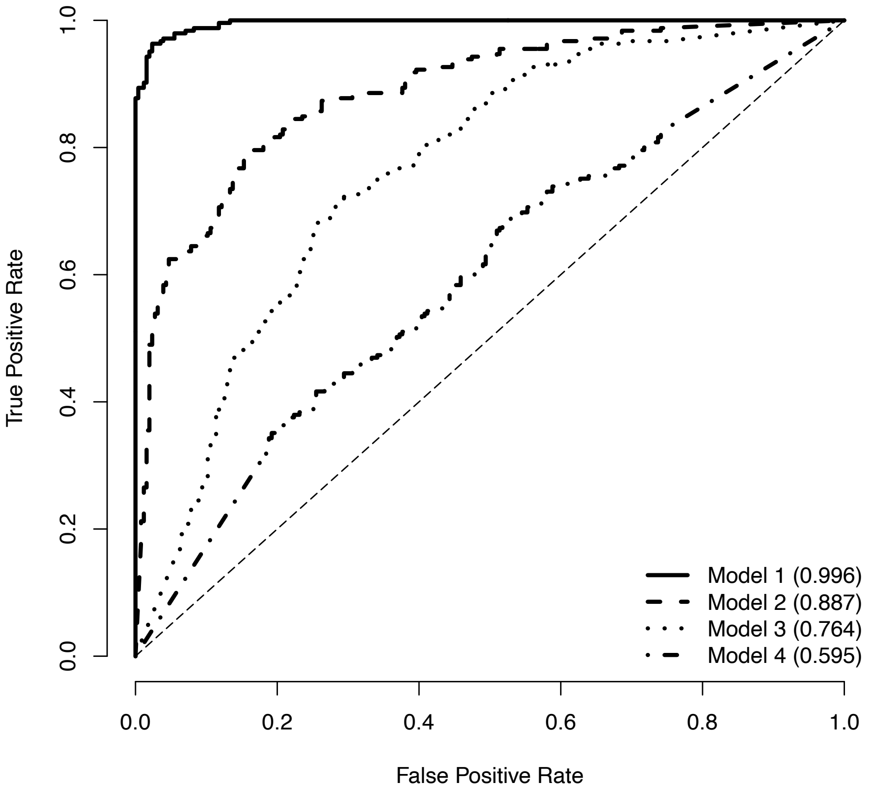


ROC Analysis

- Often ROC curves for multiple prediction models are plotted on a single ROC plot, showing easy comparison of performance



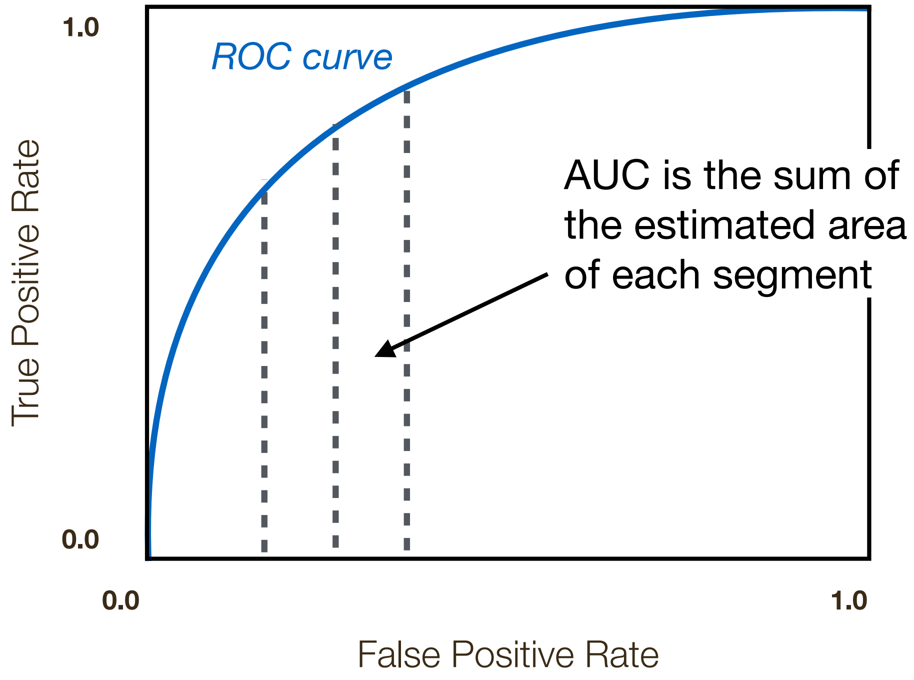
ROC curve for spam example dataset



ROC curves for 4 models on significantly larger spam dataset

AUC

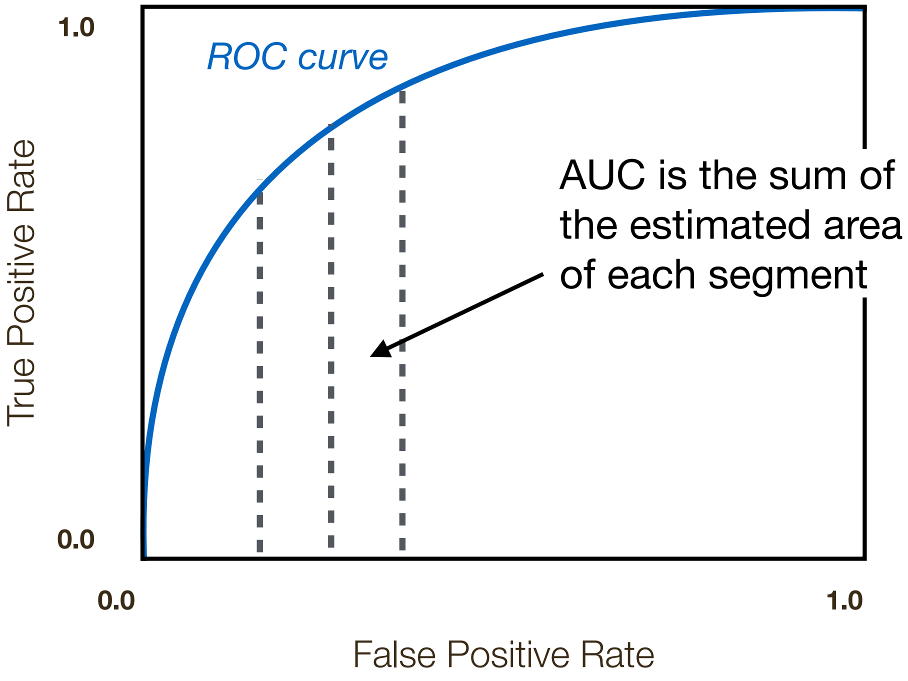
- Comparing ROC curves visually is useful but it is preferable to have a single measure to assess models
- Area Under the ROC Curve (AUC) measures the area beneath an ROC curve, bigger area is better performance
- Calculated as the integral of the ROC curve but can be done using trapezoidal method



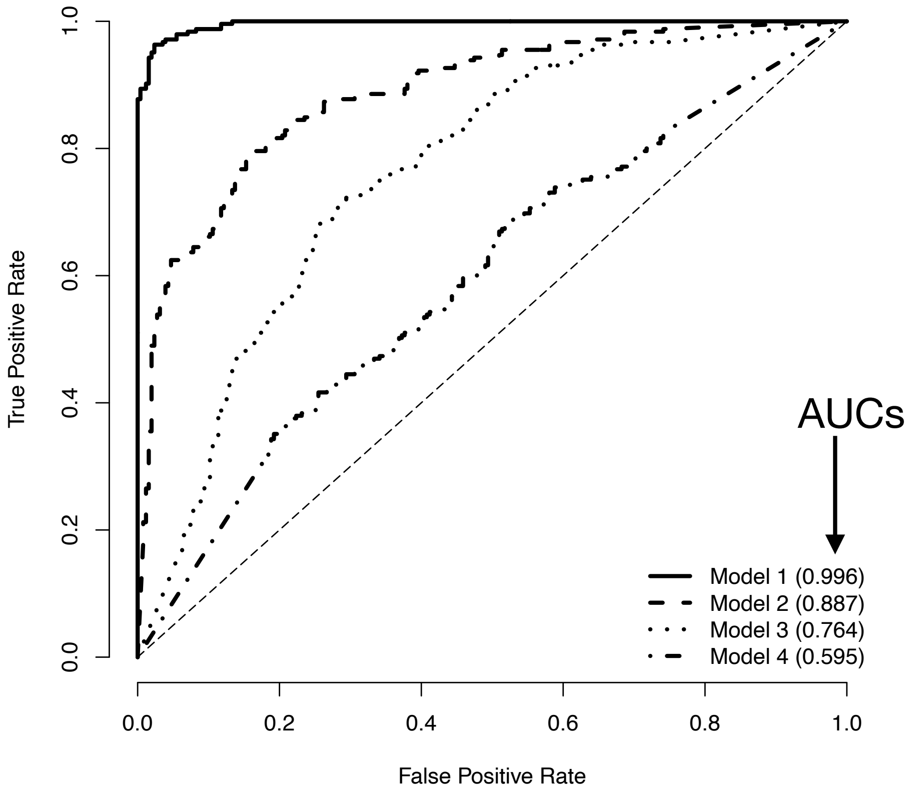
AUC is the sum of the estimated area of each segment

AUC

- Comparing ROC curves visually is useful but it is preferable to have a single measure to assess models
- Area Under the ROC Curve (AUC) measures the area beneath an ROC curve, bigger area is better performance
- Calculated as the integral of the ROC curve but can be done using trapezoidal method



AUC is the sum of the estimated area of each segment



Performance Measures - Continuous Targets

- Sum of Squared Errors (SSE)
  - Most common measure for continuous targets

$$\text{SSE} = \frac{1}{2}\sum_{i=1}^{n}\left(t_i - \hat{t_i}\right)^2$$

t_i is target value of instance i

$\hat{t_i}$ is predicted value of instance i

- Mean Squared Error (MSE)
  - average difference between predictions and actual values of the test set

$$\text{MSE} = \frac{\sum_{i=1}^{n}\left(t_i - \hat{t_i}\right)^2}{n}$$

  - Allows the ranking of performance of multiple models
  - Not meaningful in relation to the scenario the model is being used for (error not in the same units as target)

Performance Measures - Continuous Targets

- Root Mean Squared Error (RMSE)
  - in the same units as the target value so more meaningful

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} \left(t_i - \hat{t_i}\right)^2}{n}}$$

  - Tends to over emphasise individual large errors (due to square)

- Mean Absolute Error (MAE)
  - addresses the problem of using the square
    (similar to using Manhattan over Euclidean distance)

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |t_i - \hat{t_i}|}{n}$$

- But both require domain knowledge…

Performance Measures - Continuous Targets

- R^2 Coefficient is normalised domain independent measure of model performance

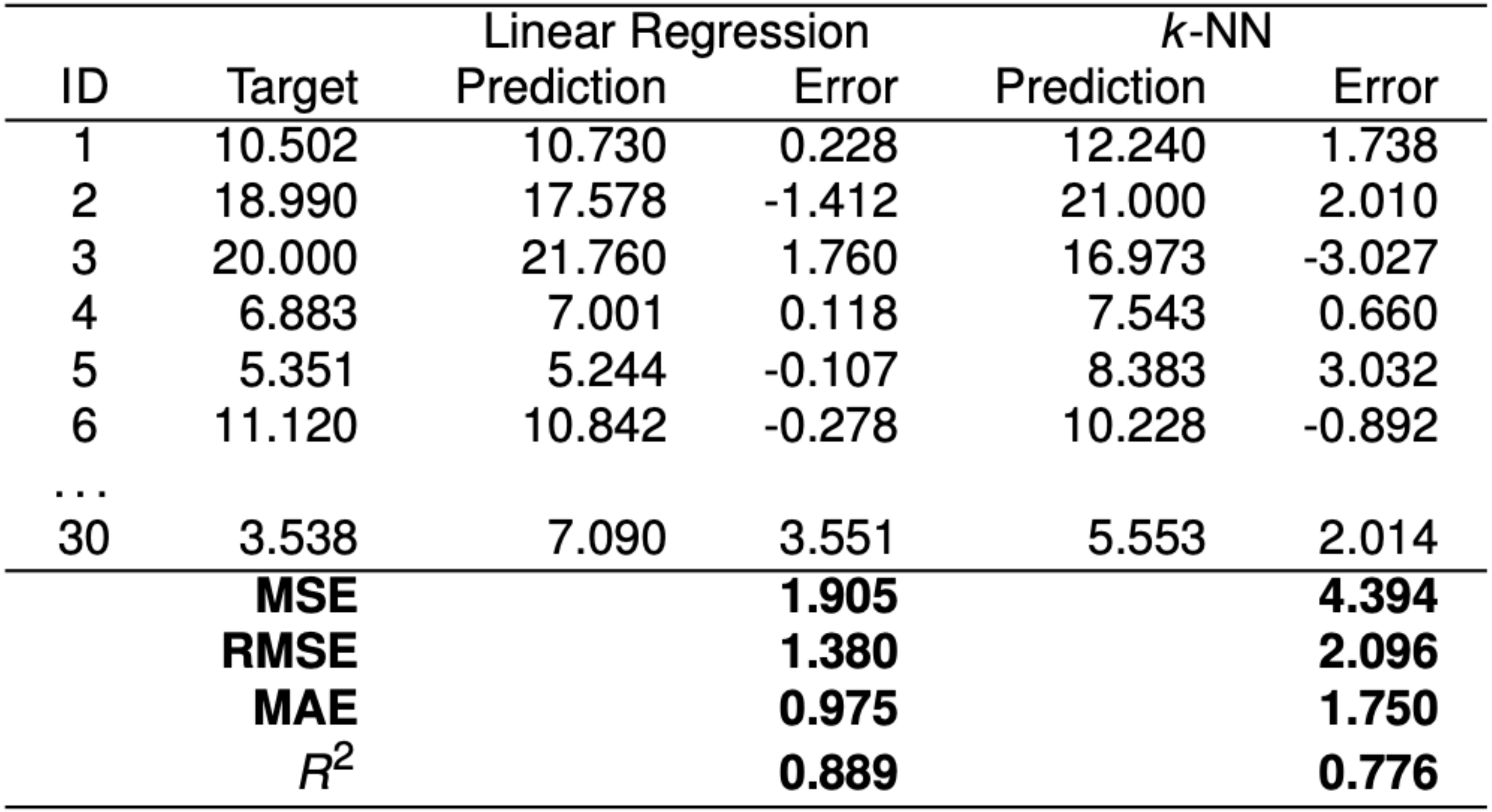
$$R^2 = 1 - \frac{\text{sum of squared errors}}{\text{total sum of squares}}$$

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

- Compares the performance of a model on a test dataset with the performance of an *imaginary* model that always predicts the average values from the test dataset
  - always in range [0,1]
  - higher numbers better

Example

- Comparing the prediction of two models for a blood-thinning drug dosage prediction problem



		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
...					
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**

What measure to use?

- Guideline:

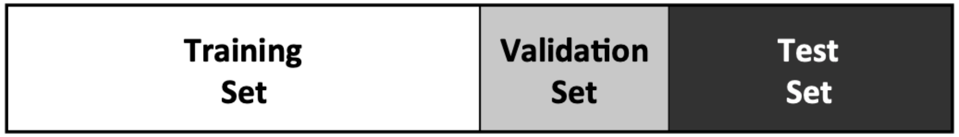
  Better to be pessimistic when evaluating performance

- Classification
  - Recommend use of Average Class Accuracy using harmonic mean

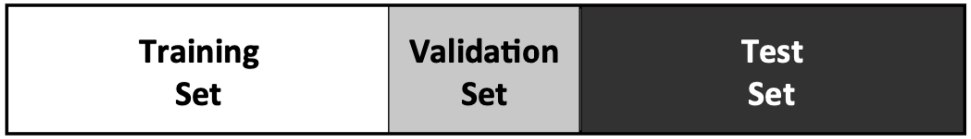
- Regression (continuous targets)
  - Recommend use of R^2 Coefficient

Designing Evaluation Experiments

- Distinct, random, non-overlapping samples of data needed for evaluation
- Hold-out sampling divides the dataset into training, validation and test sets
    - No fixed recommendations on proportions, 50:20:30 or 40:20:40 are common
- Issues:
    - Is there enough data?
    - Can be misleading with ‘lucky’ splits of the data



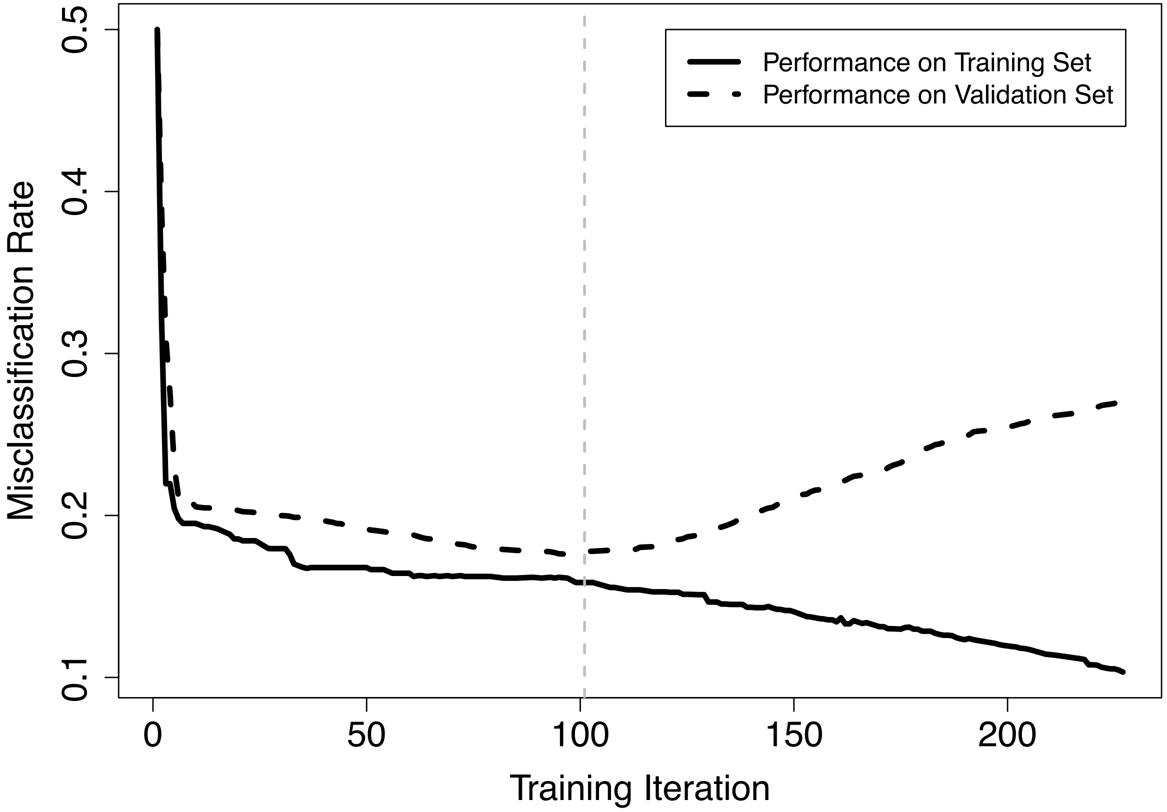
(a) A 50:20:30 split



(b) A 40:20:40 split

Validation set

- Validation set is used for tuning the model, e.g. hyerparameter fixing or feature selection,
- Also useful in iterative ML algorithms, e.g. decision trees, regression models, to avoid overfitting



Similar to post-pruning process in decision trees

Recap: Overfitting

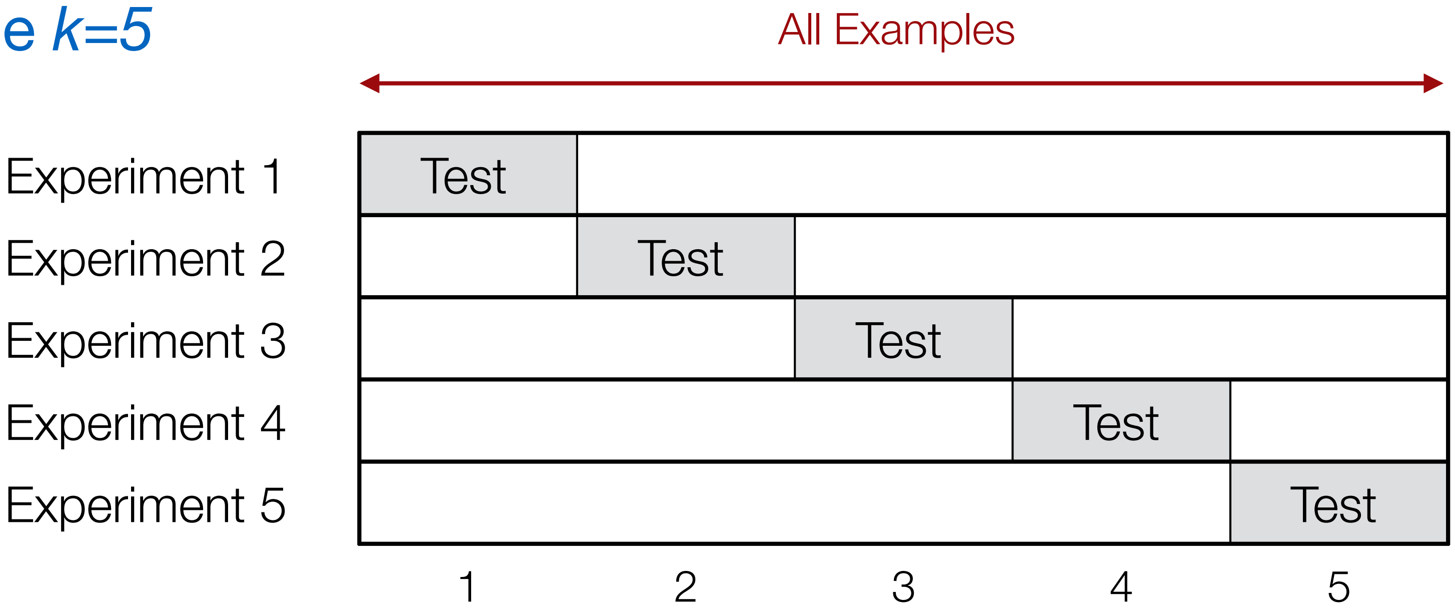
- For real-world tasks, we are interested in generalisation accuracy.
- Overfitting: Model is fitted too closely to the training data (including its noise). The model cannot generalise to situations not presented during training, so it is not useful when applied to unseen data.
- **Possible Causes**
  - *Small training set*: Classifier only given a few examples, may not be representative of the underlying concepts.
  - *Complex model*: Model has too many parameters relative to the number of training examples.
  - *Noise*: Spurious or contradictory patterns in the training data.
  - *High-dimensionality*: Data has many irrelevant features (dimensions) containing noise which leads to a poor model.

➡ A good model must not only fit the training data well, but also accurately classify examples that it has never seen before.

k-fold Cross Validation

- *k*-Fold Cross Validation
  - Divide the data into *k* disjoint subsets - “folds” (e.g. *k=5* or *10*).
  - For each of *k* experiments, use *k-1* folds for training and the selected one fold for testing.
  - Repeat for all *k* folds, average the accuracy/error rates.

Example *k=5*



All Examples

	1	2	3	4	5
Experiment 1	Test				
Experiment 2		Test			
Experiment 3			Test		
Experiment 4				Test	
Experiment 5					Test

Example: Cross Validation

- Number of correct and incorrect predictions made by a spam classifier on 300 emails, when we run 5-fold cross validation.

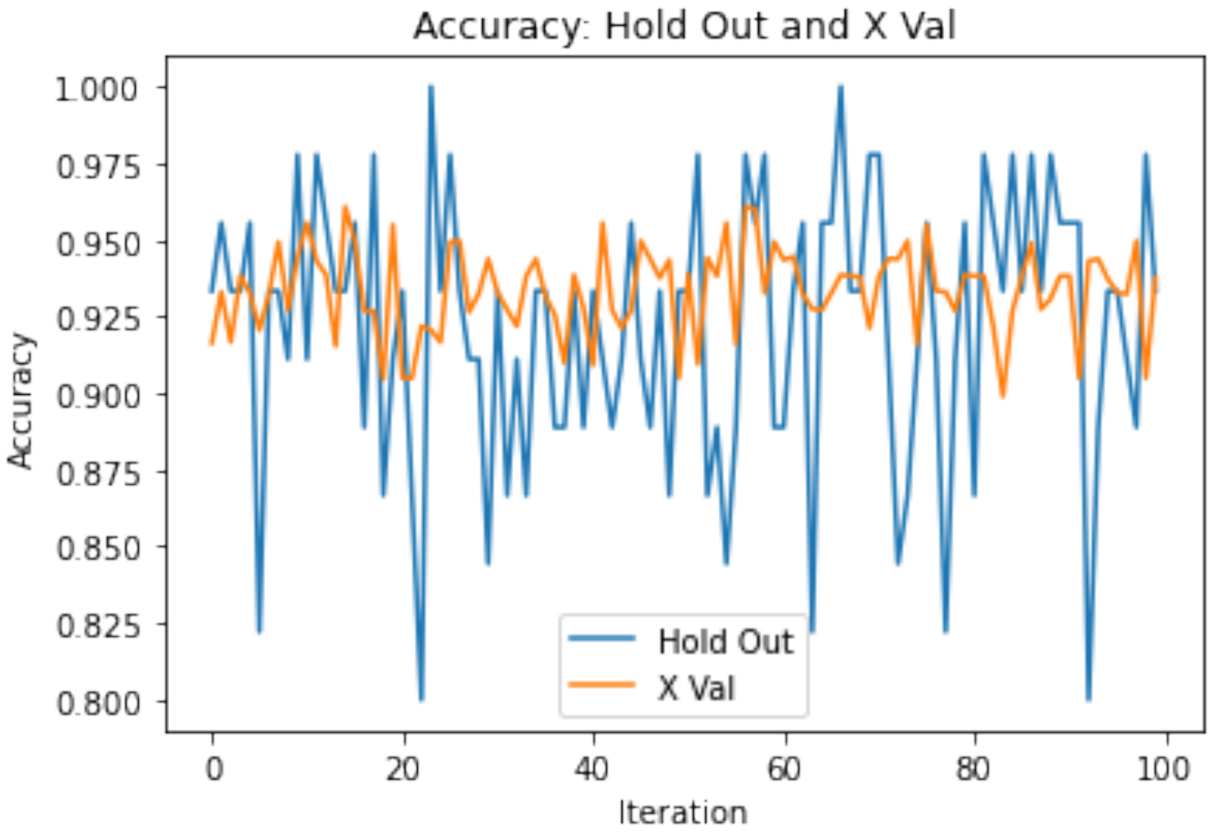
	Class: Non-Spam		**Class: Spam**		
Fold	Correct	Incorrect	Correct	Incorrect	**Accuracy**
1	173	22	87	18	86.67%
2	107	88	71	34	59.33%
3	143	52	80	25	74.33%
4	185	10	59	46	81.33%
5	162	33	71	34	77.67%
Mean	154	41	74	31	**75.87%**

Accuracy for each fold

Average accuracy across all 5 folds

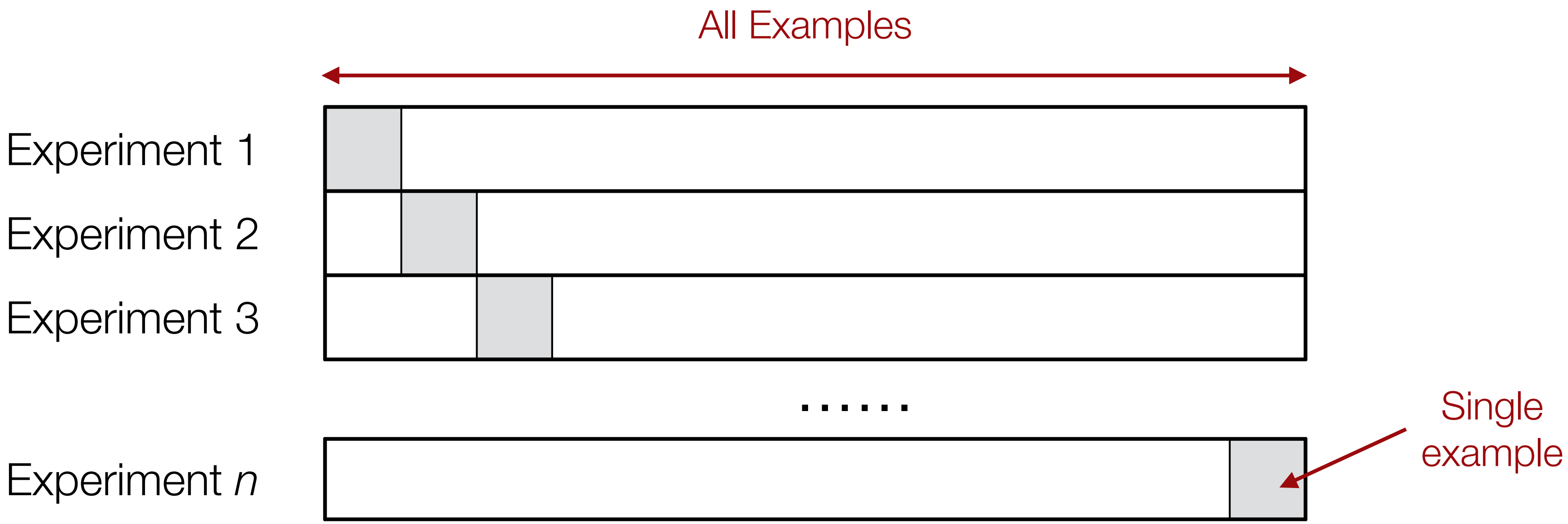
Hold-Out vs Cross Val

- Cross Validation
  - ✓ More stable estimate of accuracy
  - ✓ Better estimate of accuracy (Hold-Out can underestimate)
  - ✗ Slower



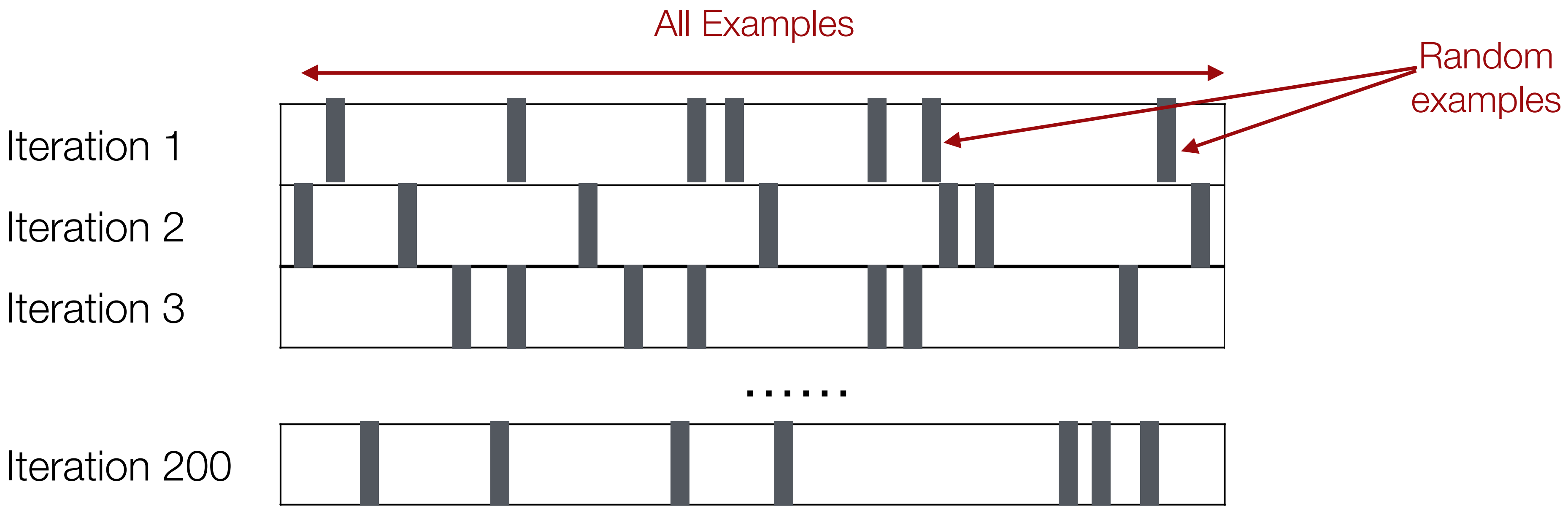
Leave-one-out Cross Validation

- Leave-one-out (LOO) Extreme case of *k*-Fold Cross Validation where *k* is selected to be the total number of examples in the dataset.
  - For a dataset with *n* examples, perform *n* experiments.
  - For each experiment use *n-1* examples for training and the remaining single example for testing.
  - Average the accuracy/error rates over all *n* experiments.



Bootstrapping

- Bootstrapping approaches are preferred over cross-validation in contexts with very small datasets (<300 examples)
  - A random selection of examples is selected for testing, remaining used for training
  - Repeat multiple times, generally significantly more than *k* in k-fold CV (>=200 iterations)
  - Average accuracy/error results across all iterations



Summary

- Choice of performance measures depends on

  A. nature of the problem, i.e. continuous vs categorical;

  B. characteristics of the dataset, i.e. balanced vs unbalanced;

  C. needs of the application, (e.g. medical diagnosis vs marketing response prediction) or domain

- Recommend in absence of other factors
  - Classification - use ACA$_{HM}$
  - Regression - use
- Data splitting approach R^2
  - Very small datasets, use Bootstrapping
  - Very large datasets, use Holdout sampling
  - Otherwise, consider Cross-Validation
  - May need to consider natural structure in the data, e.g. where there is a time dimension, use data from one period for training and another period for testing (known as out-of-time sampling)