Data Science for Business
Evaluation in Machine Learning - Part 2

Asst. Prof. Teerapong Leelanupab (Ph.D.)
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang (KMITL)

Week 10

Overview

- Part 2
  - Decision Analytic Thinking
    - Evaluation Metrics for Classification
    - Expected Value Framework
    - Evaluation Metrics for Classification
  - Visualizing Model Performance
  - Experimental Setup

Data Science for Business
Decision Analytic Thinking

Asst. Prof. Teerapong Leelanupab (Ph.D.)
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang (KMITL)

Week
10.1

Decision Analytic Thinking

- **Evaluation Metrics for Classification**
- Expected Value Framework
- Evaluation Metrics for Regression

Why de we need to evaluation?

Essentially, all models are wrong, but some are useful.

—George E. P. Box

- Allow systematic and objective comparison between different machine learning models in prediction.
- Most common *machine learning* tasks are:
  - Classification
  - Regression

Evaluation

- How do we measure generalization performance?

Evaluating Classifiers: Plain Accuracy

$$\text{Accuracy} = \frac{\text{Number of correct decisions made}}{\text{Total number of decisions made}}$$

$$= 1 - \text{error rate}$$

- *Too* simplistic..

Evaluation

When making predictions, we need some kind of measure to capture how often the model makes correct or incorrect predictions, and how severe the mistakes are.

Misclassification Rate:
Fraction of incorrect predictions made by the classifier.

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

Accuracy:
Fraction of correct predictions made by the classifier.

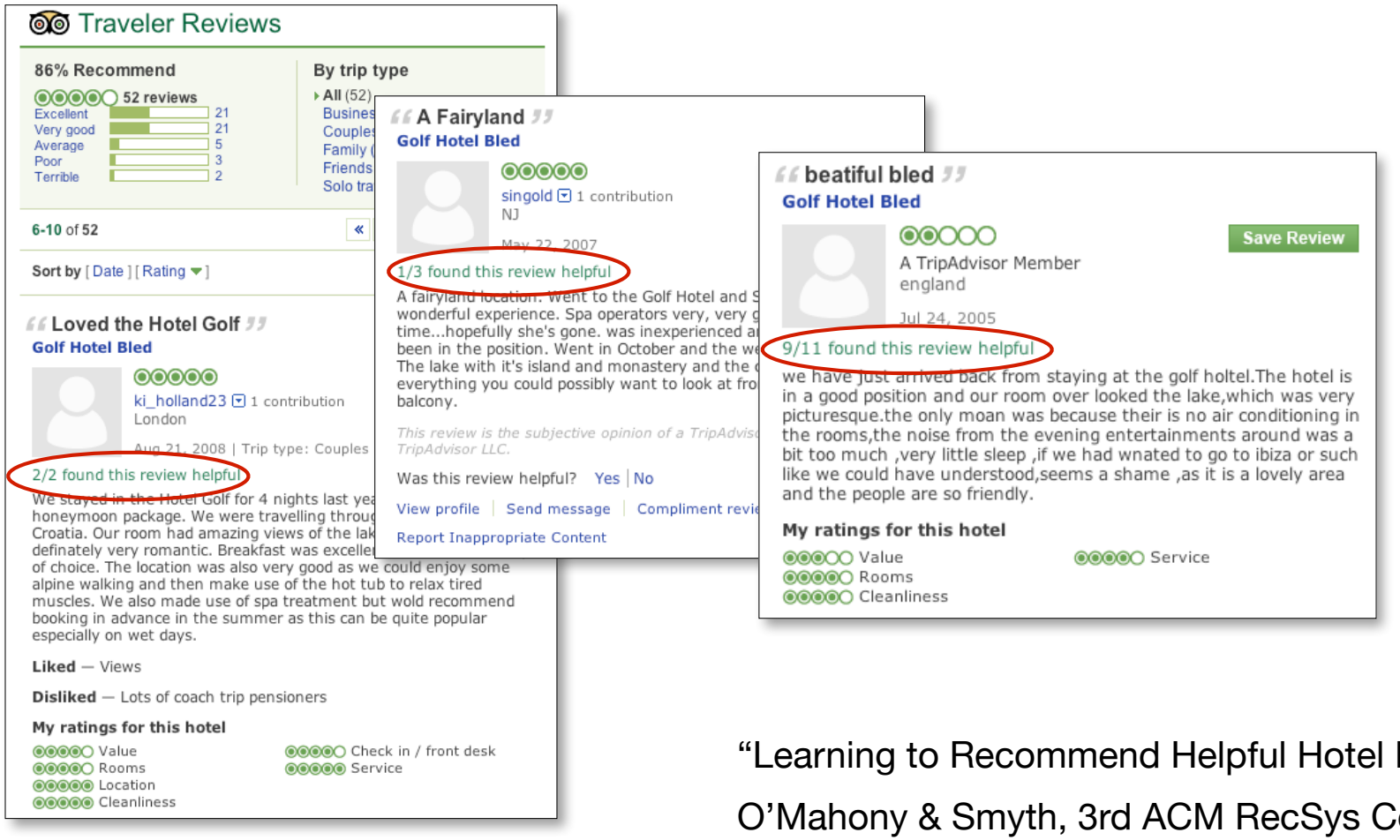
$$ACC = \frac{\#\text{ correct predictions}}{\text{total predictions}}$$

Email	Label	Prediction	Correct?
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	✓

$$MR = \frac{3}{10} = 0.3 \qquad ACC = \frac{7}{10} = 0.7$$

Example: Hotel Reviews

Q. Can we predict the “helpfulness” of TripAdvisor hotel reviews?



“Learning to Recommend Helpful Hotel Reviews”

O’Mahony & Smyth, 3rd ACM RecSys Conference, 2009

http://dl.acm.org/citation.cfm?id=1639774

Example: Hotel Reviews

- Compare performance of Naïve Bayes and Support Vector Machine (SVM) classifiers on review data using Weka.
- Test set: 105 “Helpful”, 60 “Unhelpful” reviews
- Testing option: Hold-out validation with 66/33% hold-out split.

SVM

```
Correctly Classified Instances         117               70.9091 %
Incorrectly Classified Instances        48               29.0909 %
Kappa statistic                          0.3071
Mean absolute error                      0.2909
Root mean squared error                  0.5394
Relative absolute error                 62.6804 %
Root relative squared error            112.1168 %
Total Number of Instances              165

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.895    0.617      0.718     0.895      0.797      0.639    good
                 0.383    0.105      0.676     0.383      0.489      0.639    bad
Weighted Avg.    0.709    0.431      0.703     0.709      0.685      0.639

=== Confusion Matrix ===

  a  b   <-- classified as
 94 11 |  a = good
 37 23 |  b = bad
```

Naïve Bayes

```
Correctly Classified Instances         103               62.4242 %
Incorrectly Classified Instances        62               37.5758 %
Kappa statistic                          0.1995
Mean absolute error                      0.3793
Root mean squared error                  0.5316
Relative absolute error                 81.7353 %
Root relative squared error            110.5048 %
Total Number of Instances              165

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.686    0.483      0.713     0.686      0.699      0.674    good
                 0.517    0.314      0.484     0.517      0.5        0.674    bad
Weighted Avg.    0.624    0.422      0.63      0.624      0.627      0.674

=== Confusion Matrix ===

  a  b   <-- classified as
 72 33 |  a = good
 29 31 |  b = bad
```

Example: Hotel Reviews

- Compare performance of Naïve Bayes and Support Vector Machine (SVM) classifiers on review data using Weka.
- Test set: 105 “Helpful”, 60 “Unhelpful” reviews
- Testing option: Hold-out validation with 66/33% hold-out split.

SVM

Accuracy = 70.9%

	Prediction: H	Prediction: U
Real World: H	94	11
Real World: U	37	23

SVM biased toward majority class (H)

Naïve Bayes

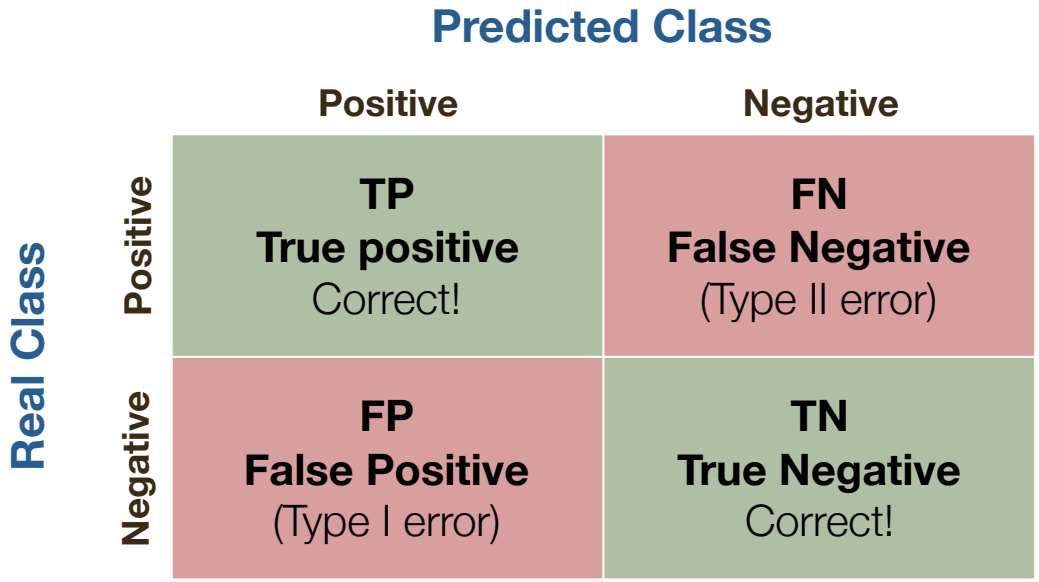
Accuracy = 62.4%

	Prediction: H	Prediction: U
Real World: H	72	33
Real World: U	29	31

What if this is important?

Evaluation Measures

Confusion matrix summarises the performance of an algorithm, when compared with the real classes ("ground truth").



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

Building a Confusion Matrix

Default Truth	Model Prediction
0	0
1	1
0	1
0	1
0	0
1	1
0	0
0	0
1	1
1	0

Predicted class Actual class	Default	No Default	Total
Default	3	1	4
No Default	2	4	6
Total	5	5	10

Evaluation Measures

Spam Filtering Example: Predict emails as *spam* or *non-spam*…

- **TP** = Spam emails correctly predicted as spam
- **FP** = Non-spam emails incorrectly predicted as spam
- **TN** = Non-spam emails correctly predicted as non-spam
- **FN** = Spam emails incorrectly predicted as non-spam

Email	Label	Prediction	Correct?	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
FP=2	**TN=3**	Non

Real Class

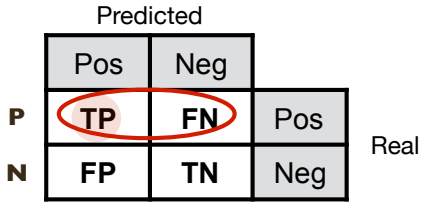
Evaluation Measures

- Accuracy: Fraction of predictions correct

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

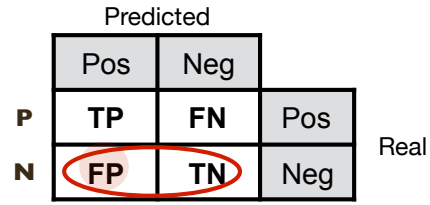
- True Positive Rate:
  Focus on TPs
  Also called *Sensitivity* ความสามารถในการทำนายคลาสๆ หนึ่ง คลาสถูก)
  or called *Recall*, aka. Probability of Detection

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



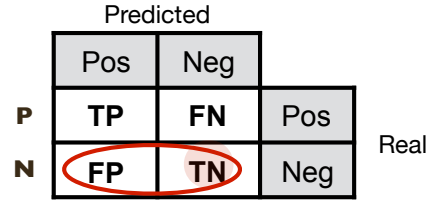
- False Positive Rate:
  Focus on FPs

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



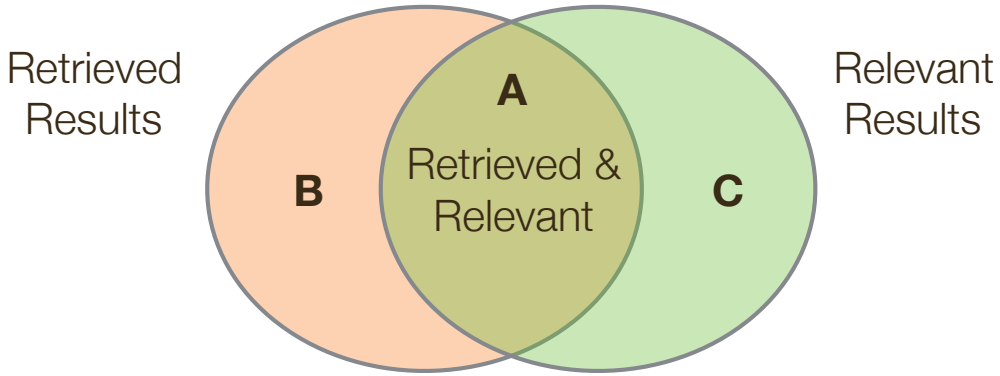
- True Negative Rate:
  Focus on TNs
  Also called *Specificity*

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



Precision & Recall

- Measures from information retrieval, but used in ML evaluation.
- Precision: proportion of retrieved results that are relevant. โมเดลทำนาย คลาสถูกได้ดีขนาดไหน
- 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 “water charges”. In fact, 45 relevant documents actually exist.

Perform a search, 9 out of 10 results on first page are relevant documents.

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

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

Precision & Recall

- Precision & Recall also used as general measures to evaluate machine learning algorithms.

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

เรา sensitive ต้อ
อะไร(FP, FN) เราก็ค่อยไป
หา recall หรือ precision

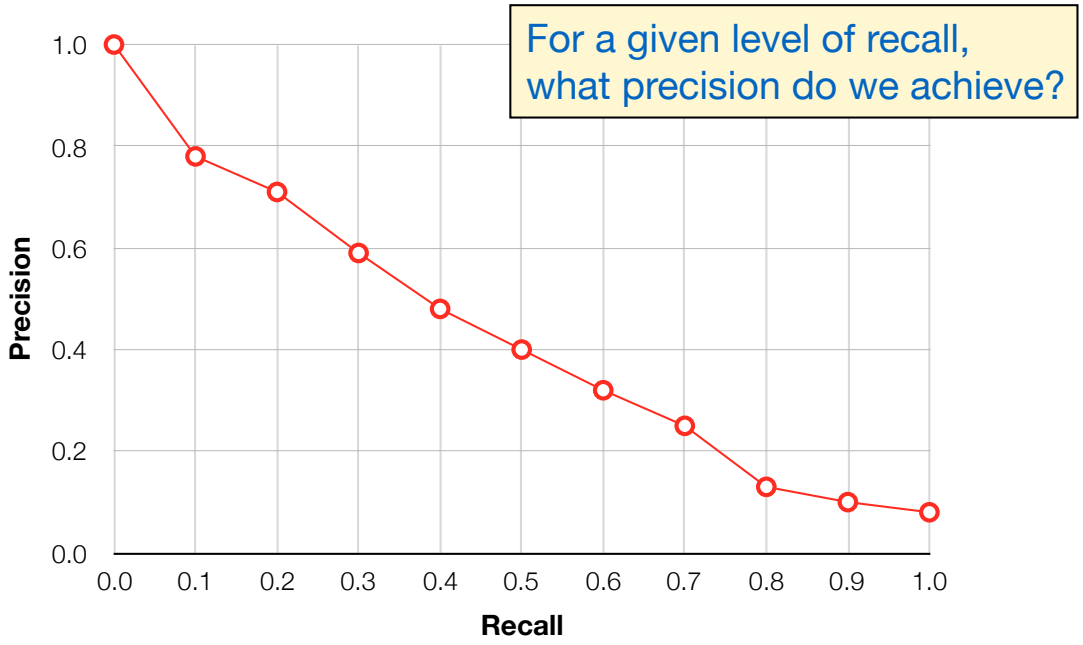
		Predicted: Pos	Predicted: Neg	
P	Real	TP	FN	Pos
N	Real	FP	TN	Neg

$$\text{Recall} = \frac{TP}{TP + FN} = \text{Sensitivity}$$

มือถือ

		Predicted: Pos	Predicted: Neg	
P	Real	TP	FN	Pos
N	Real	FP	TN	Neg

- 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.



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	Correct?	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
FP=2	**TN=3**	Non

Real Class

Balanced Accuracy

- Skewed class distributions occur when one class is over-represented in the data.
  - e.g. Fraud detection: Vast majority of financial transactions are legitimate, a small fraction are fraudulent.
  - Other examples: medical diagnosis, e-commerce, security.
- High accuracy can be achieved by biased (or trivial) classifiers.

$$\Rightarrow \text{Accuracy} = 90\%$$

	Classified as: Pos	Classified as: Neg
Real World: Pos	**0**	**10**
Real World: Neg	**0**	**90**

- To deal with skewed classes, use a balanced evaluation measure. Measures include:
  - **Balanced Accuracy Rate (BAR):** Mean of TP Rate and TN Rate
  - **Balanced Error Rate (BER):** Mean of FP Rate and FN Rate

Balanced Accuracy and Error Rate

Balanced Accuracy Rate (BAR)

$$\text{Balanced Accuracy (Rate)} = \frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{FP+TN}\right)$$

balance ระหว่าง recall กับ TN rate

Balanced Error Rate (BER)

$$\text{Balanced Error Rate} = \frac{1}{2}\left(\frac{FP}{FP+TN} + \frac{FN}{TP+FN}\right)$$

https://en.wikipedia.org/wiki/Evaluation_of_binary_classifiers
http://www.modelselect.inf.ethz.ch/evaluation.php

Combine and Weight Precision and Recall

- F-Measure: A single measure that trades off precision against recall, for a given level of balance.

$$\mathrm{F} = \frac{(1+\beta^2) \times \mathrm{Precision} \times \mathrm{Recall}}{\beta^2 \times \mathrm{Precision} + \mathrm{Recall}}$$

- The Beta parameter controls the trade-off:
  - $\beta < 1$ Focus more on Precision
  - $\beta = 1$ Harmonic mean of Precision and Recall.
  - $\beta > 1$ Focus more on Recall

- F1-Measure: Most widely-used variant, sets $\beta = 1$

อยากเขียน recall กับ precision ให้อยู่ในสูตรเดียว (เห็นทั้ง 2) โดยมี Beta มาเป็น parameter

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

harmonic mean of precision and recall

Decision Analytic Thinking

- Evaluation Metrics for Classification
- **Expected Value Framework**
- Evaluation Metrics for Regression

Expected Value Framework

A Key Analytical Framework: Expected Value

- The expected value computation provides a framework that is useful in organizing thinking about data-analytic problems
- It decomposes data-analytic thinking into:
  - the structure of the problem,
  - the elements of the analysis that can be extracted from the data, and
  - the elements of the analysis that need to be acquired from other sources
- The general form of an expected value calculation:
- $EV = p(o_1) \times v(o_1) + p(o_2) \times v(o_2) + p(o_3) \times v(o_3) + ..$

Expected Value Framework in Use Phase

- **Online marketing:**
- Expected benefit of targeting $= p_R(x) \times v_R + [1 - p_R(x)] \times v_{NR}$
- Product Price: $200
- Product Cost: $100
- Targeting Cost: $1

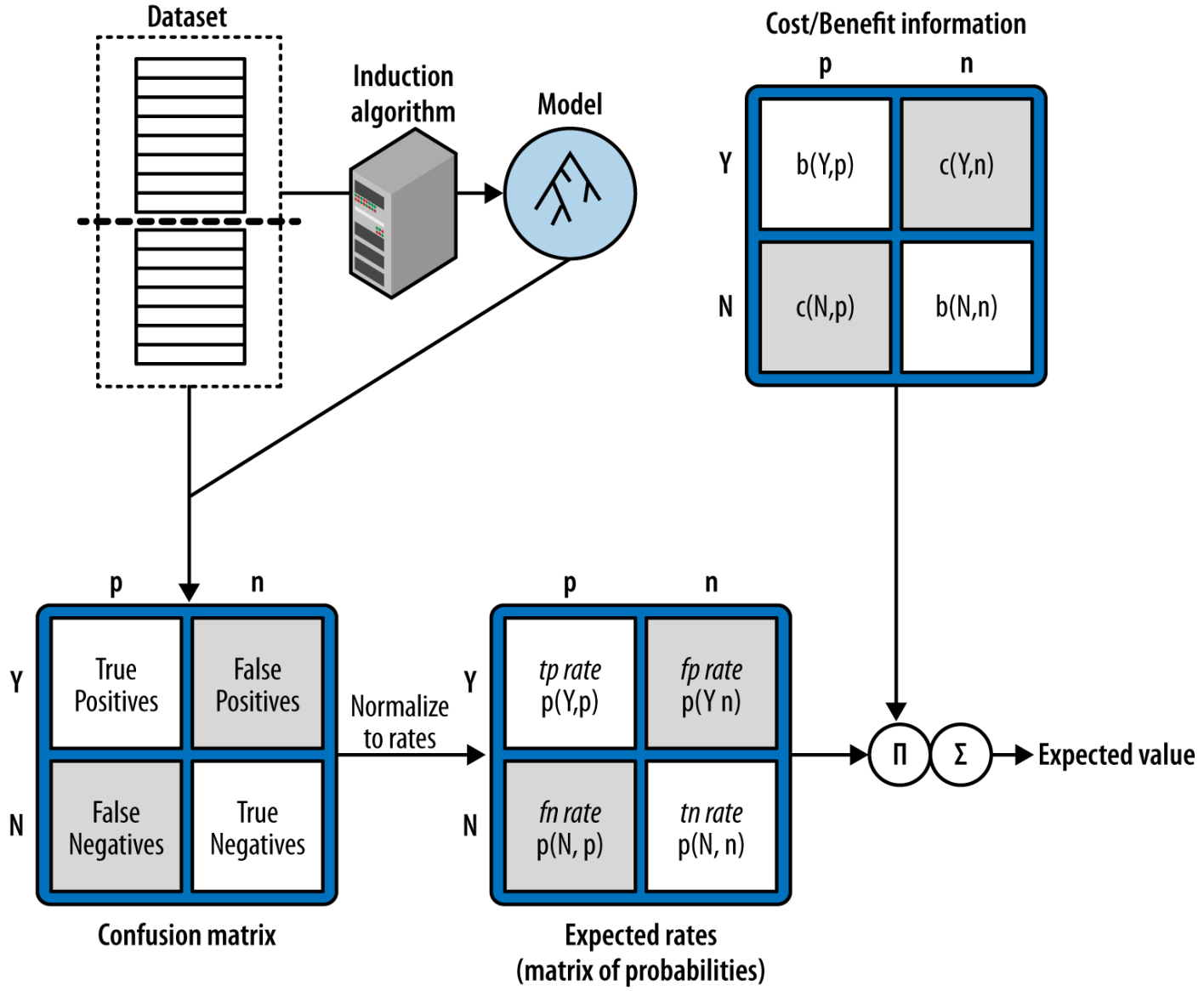
$$p_R(\boldsymbol{x}) \times \$99 - [1 - p_R(\boldsymbol{x})] \times \$1 > 0$$

$$p_R(\boldsymbol{x}) > 0.01$$

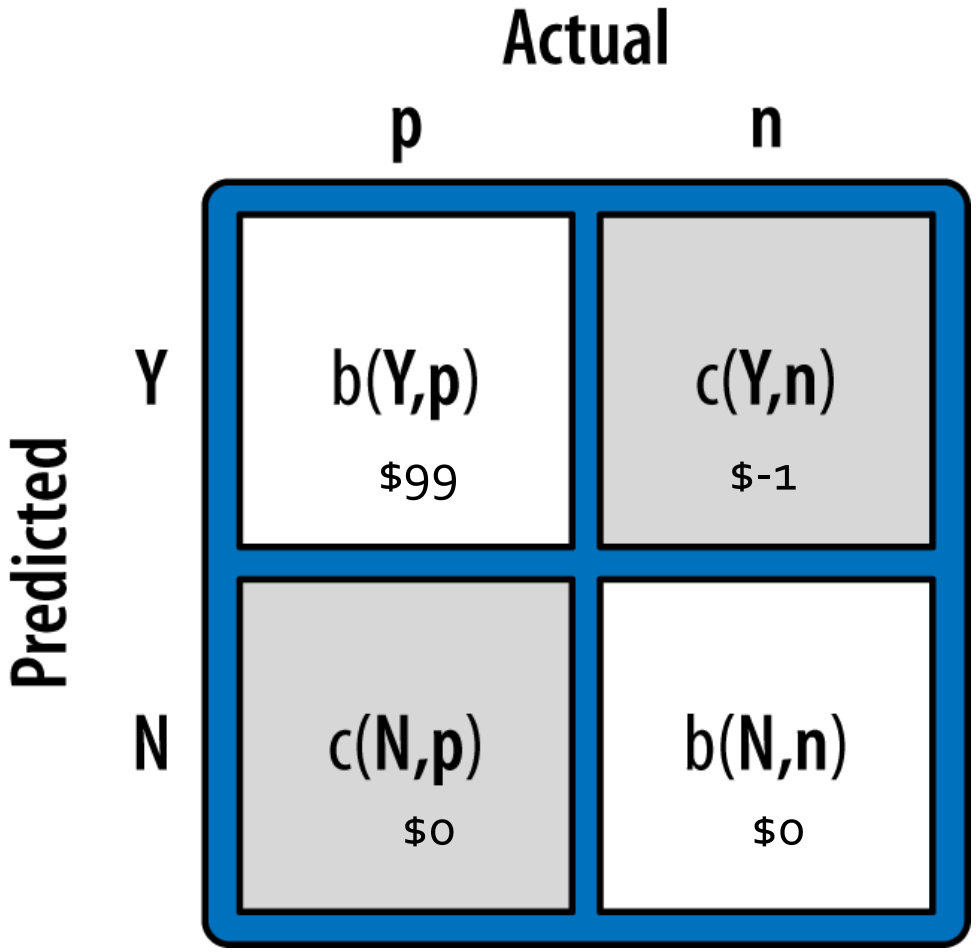
เราไม่รู้

PR(X)*99 + [-1-PR(X)]*(-1)

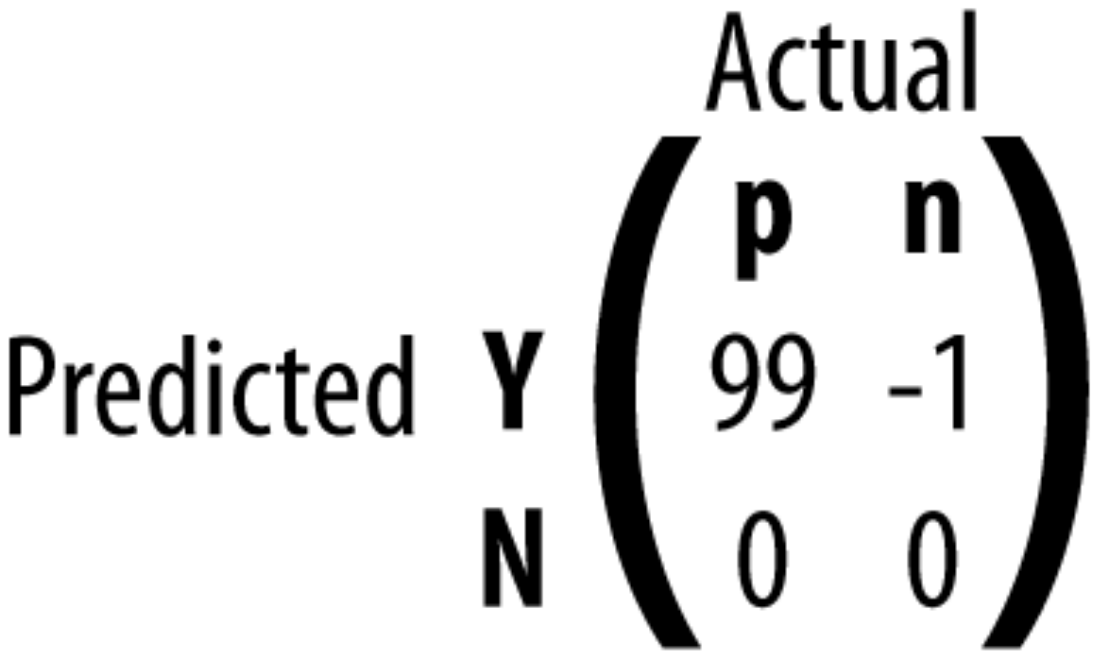
Using Expected Value to Frame Classifier Evaluation



A cost-benefit matrix



A cost-benefit matrix for the marketing example



Conditional Probability

- A rule of basic probability is:

$$p(x, y) = p(y) \times p(x \mid y)$$

Using Expected Value to Frame Classifier Evaluation

Expected profit

$$= p(Y,p)\times b(Y,p) + p(N,p)\times b(N,p) + p(N,n)\times b(N,n) + p(Y,n)\times b(Y,n)$$

Expected profit

$$= p(Y|p)\times p(p)\times b(Y,p) + p(N|p)\times p(p)\times b(N,p) + p(N|n)\times p(n)\times b(N,n) + p(Y|n)\times p(n)\times b(Y,n)$$

Expected profit

$$= p(p)\times[p(Y|p)\times b(Y,p) + p(N|p)\times b(N,p)] + p(n)\times[p(N|n)\times b(N,n) + p(Y|n)\times b(Y,n)]$$

Using Expected Value to Frame Classifier Evaluation

$T = 110$

$P = 61$ $\qquad$ $N = 49$

$p(p) = 0.55$ $\qquad$ $p(n) = 0.45$

$p(Y|p) = 56/61 = 0.92$ $\qquad$ $p(Y|n) = 7/49 = 0.14$

$p(N|p) = 5/61 = 0.08$ $\qquad$ $p(N|n) = 42/49 = 0.86$

	p	n
Y	56	7
N	5	42

$$\text{Expected profit} = p(\boldsymbol{p})\times[p(\boldsymbol{Y}|\boldsymbol{p})\times b(\boldsymbol{Y},\boldsymbol{p}) + p(\boldsymbol{N}|\boldsymbol{p})\times b(\boldsymbol{N},\boldsymbol{p})] + p(\boldsymbol{n})\times[p(\boldsymbol{N}|\boldsymbol{n})\times b(\boldsymbol{N},\boldsymbol{n}) + p(\boldsymbol{Y}|\boldsymbol{n})\times b(\boldsymbol{Y},\boldsymbol{n})]$$

cost benefit

$$= 0.55\times[0.92\times b(\boldsymbol{Y},\boldsymbol{p}) + 0.08\times b(\boldsymbol{N},\boldsymbol{p})] + 0.45\times[0.86\times b(\boldsymbol{N},\boldsymbol{n}) + 0.14\times b(\boldsymbol{Y},\boldsymbol{n})]$$

$$= 0.55\times[0.92\times 99 + 0.08\times 0] + 0.45\times[0.86\times 0 + 0.14\times(-1)]$$

$$= 50.1 - 0.063 \approx \$\mathbf{50.04}$$

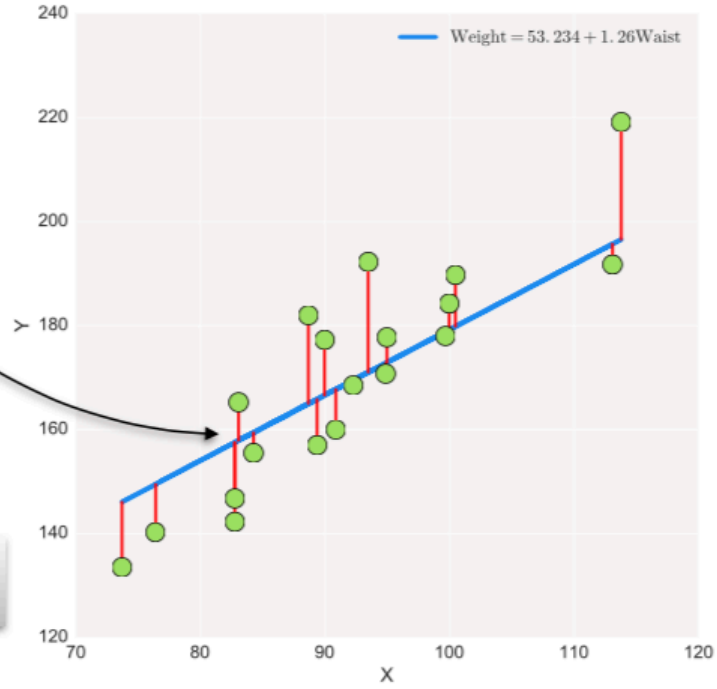
Decision Analytic Thinking

- Evaluation Metrics for Classification
- Expected Value Framework
- **Evaluation Metrics for Regression**

Key Idea is to measure the error

draw a line through the data and use a measure which gives us information on how well that line represents the data

residuals



sum(squared error) = 2333.0325

Evaluation Metrics for Regression

The coefficient of determination, R^2 มีความเป็น linear มากน้อยขนาดไหน (0 - 1) ยิ่งเยอะ ยิ่งมีความเป็น linear เยอะ

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(observed_i - predicted_i)^2}{\sum_{i=1}^{N}(predicted_i - \overline{predicted})^2}$$

Mean Squared Error (MSE)

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(observed_i - predicted_i)^2$$

Mean Absolute Error (MAE)

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|observed_i - predicted_i|$$

Mean Absolute Deviation (MAD)

$$MAD = \frac{1}{N}\sum_{i=1}^{N}|observed_i - \overline{predicted}|$$

Data Science for Business
Visualizing Model Performance

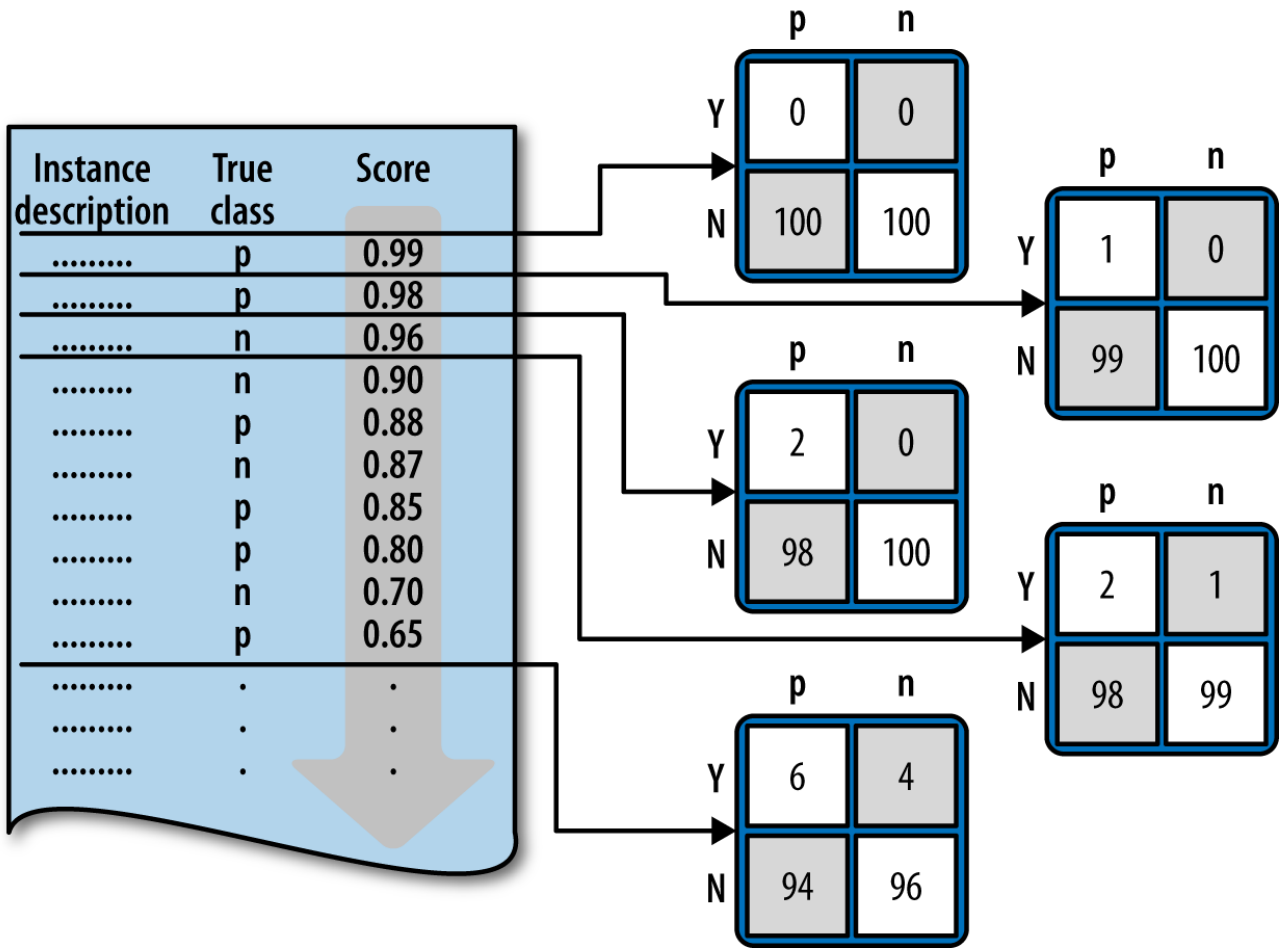
Asst. Prof. Teerapong Leelanupab (Ph.D.)
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang (KMITL)

Week
10.2

Visualizing Model Performance

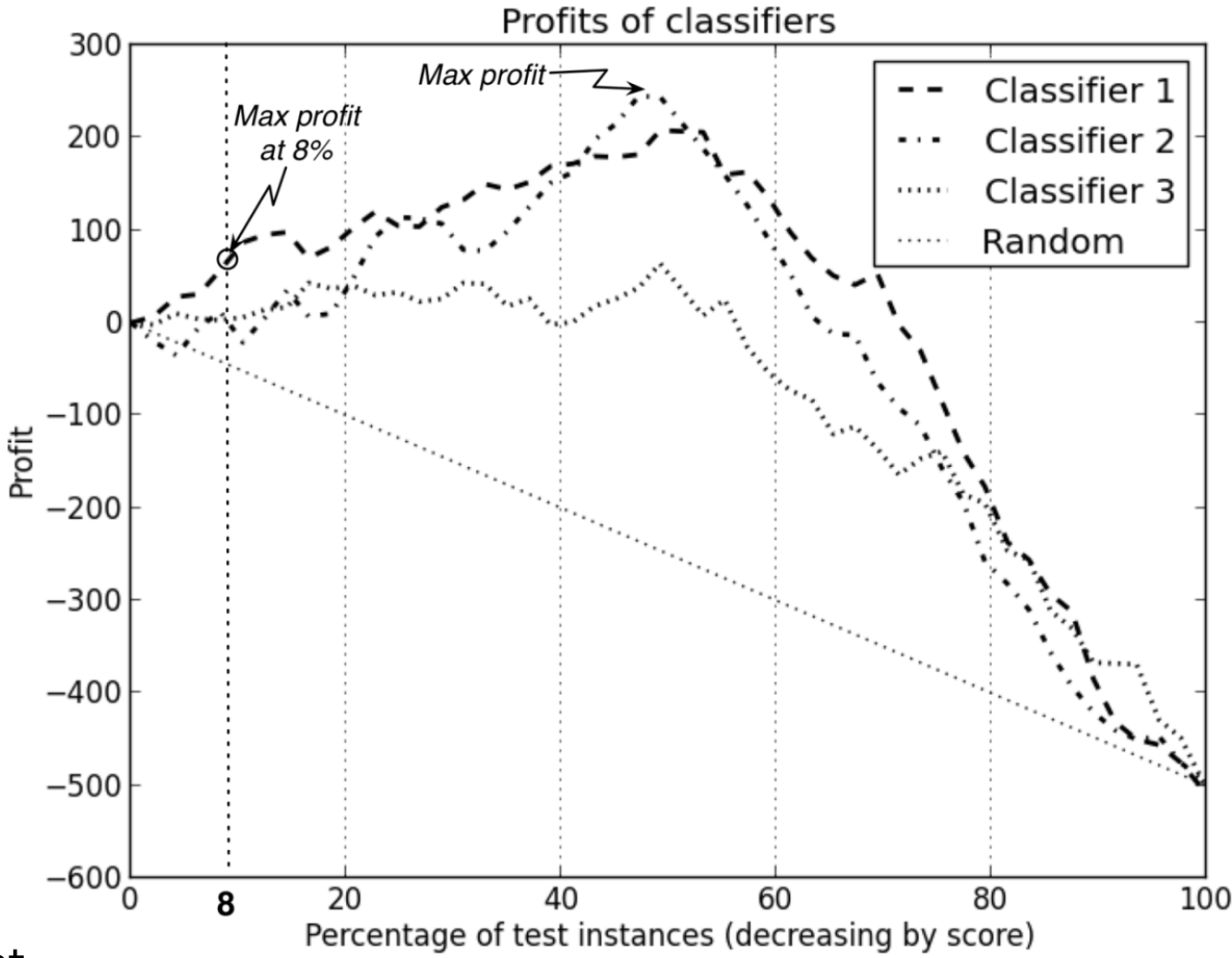
- **Profit Curves**
- Receiver Operating Characteristic (ROC Curves)
- Area Under the ROC Curve (AUC)
- Cumulative Response curve and Lift curve

Ranking Instead of Classifying



Profit Curves

Expected cumulative profit



The percentage of the population is targeted

- A test set of 100 customers, whom we test-market to prevent them from churning.
- Customers are ordered from highest to lowest probability of accepting an offer.
- Each offer costs $5 to make and market, and each accepted offer earns $9, for a profit of $4.

	p	n
Y	$4	-$5
N	$0	$0

The cost matrix

Profit Curves

- There are two critical conditions underlying the profit calculation:
- The class priors
  - The proportion of positive and negative instances in the target population
- The costs and benefits
  - The expected profit is specifically sensitive to the relative levels of costs and benefits for the different cells of the cost-benefit matrix

- *Reality..???*

Visualizing Model Performance

- Profit Curves
- **Receiver Operating Characteristic (ROC Curves)**
- Area Under the ROC Curve (AUC)
- Cumulative Response curve and Lift curve

ROC Analysis

- Many classifiers produce a prediction score, which is then converted to a binary decision based on some decision threshold θ
- Changing this threshold value can lead to different predictions, and so a different confusion matrix.

	Label	Score	> 0.5?	Prediction	Outcome
1	spam	0.1	N	non-spam	FN
2	spam	0.8	Y	spam	TP
3	non-spam	0.6	Y	spam	FP
4	spam	0.9	Y	spam	TP
5	non-spam	0.8	Y	spam	FP
6	non-spam	0.2	N	non-spam	TN
7	spam	0.8	Y	spam	TP
8	non-spam	0.6	Y	spam	FP
9	non-spam	0.1	N	non-spam	TN
10	spam	0.9	Y	spam	TP

Decision Threshold

$\theta = 0.5$

Predicted Class

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

Real Class

	Label	Score	> 0.7?	Prediction	Outcome
1	spam	0.1	N	non-spam	FN
2	spam	0.8	Y	spam	TP
3	non-spam	0.6	N	non-spam	TN
4	spam	0.9	Y	spam	TP
5	non-spam	0.8	Y	spam	FP
6	non-spam	0.2	N	non-spam	TN
7	spam	0.8	Y	spam	TP
8	non-spam	0.6	N	non-spam	TN
9	non-spam	0.1	N	non-spam	TN
10	spam	0.9	Y	spam	TP

Decision Threshold

$\theta = 0.7$

Predicted Class

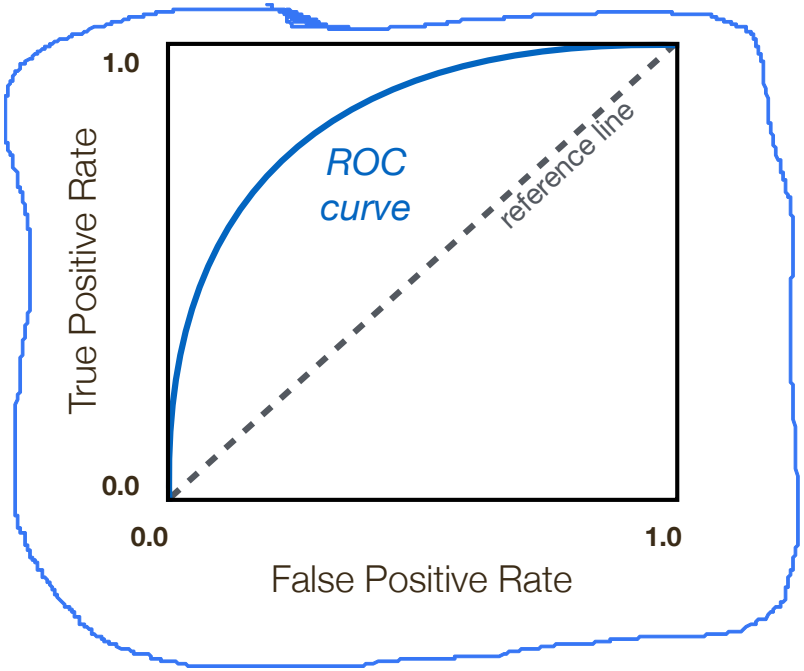
	Spam	Non
Spam	**TP=4**	**FN=1**
Non	**FP=1**	**TN=4**

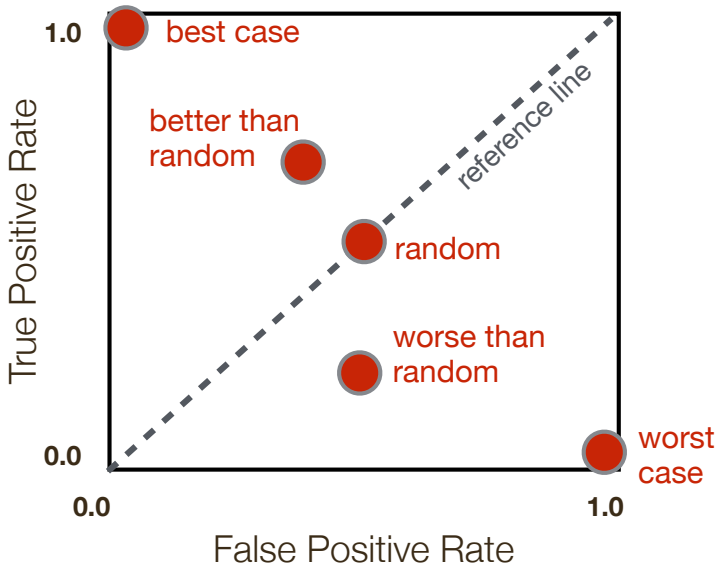
Real Class

positive rate = recall

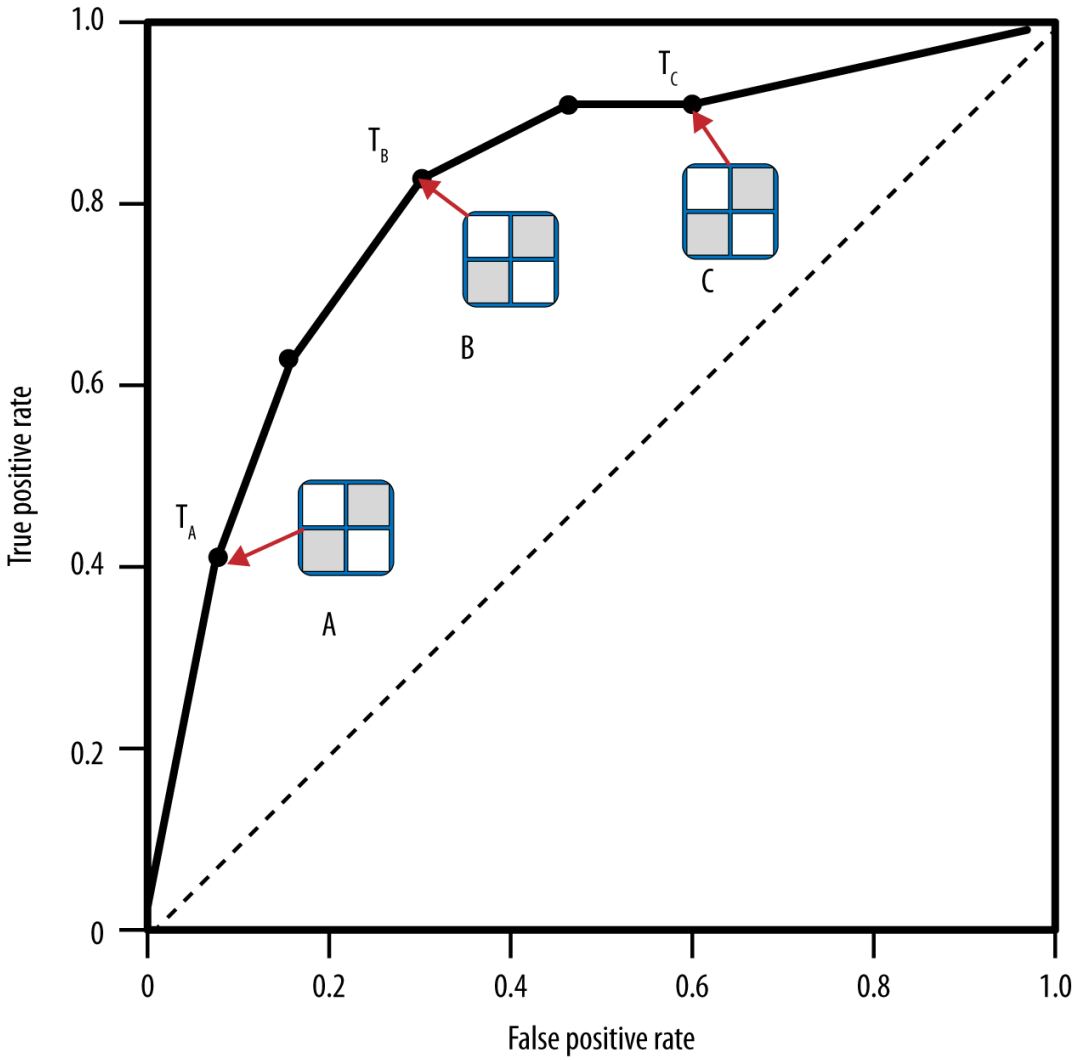
ROC Analysis

- Often want to compare the performance of classifiers at many different decision thresholds (i.e. summarise many confusion matrices).
- A Receiver Operating Characteristic (ROC Curve) is a graphical plot of how the true positive rate and false positive rate change over many different thresholds. The curve is drawn by plotting a point for each feasible threshold and joining them.
- A trained classifier should always be above the “random” reference line. The strength of the classifier increases as the ROC curve moves further from the line (i.e. closer to top left corner).





ROC Graphs and Curves

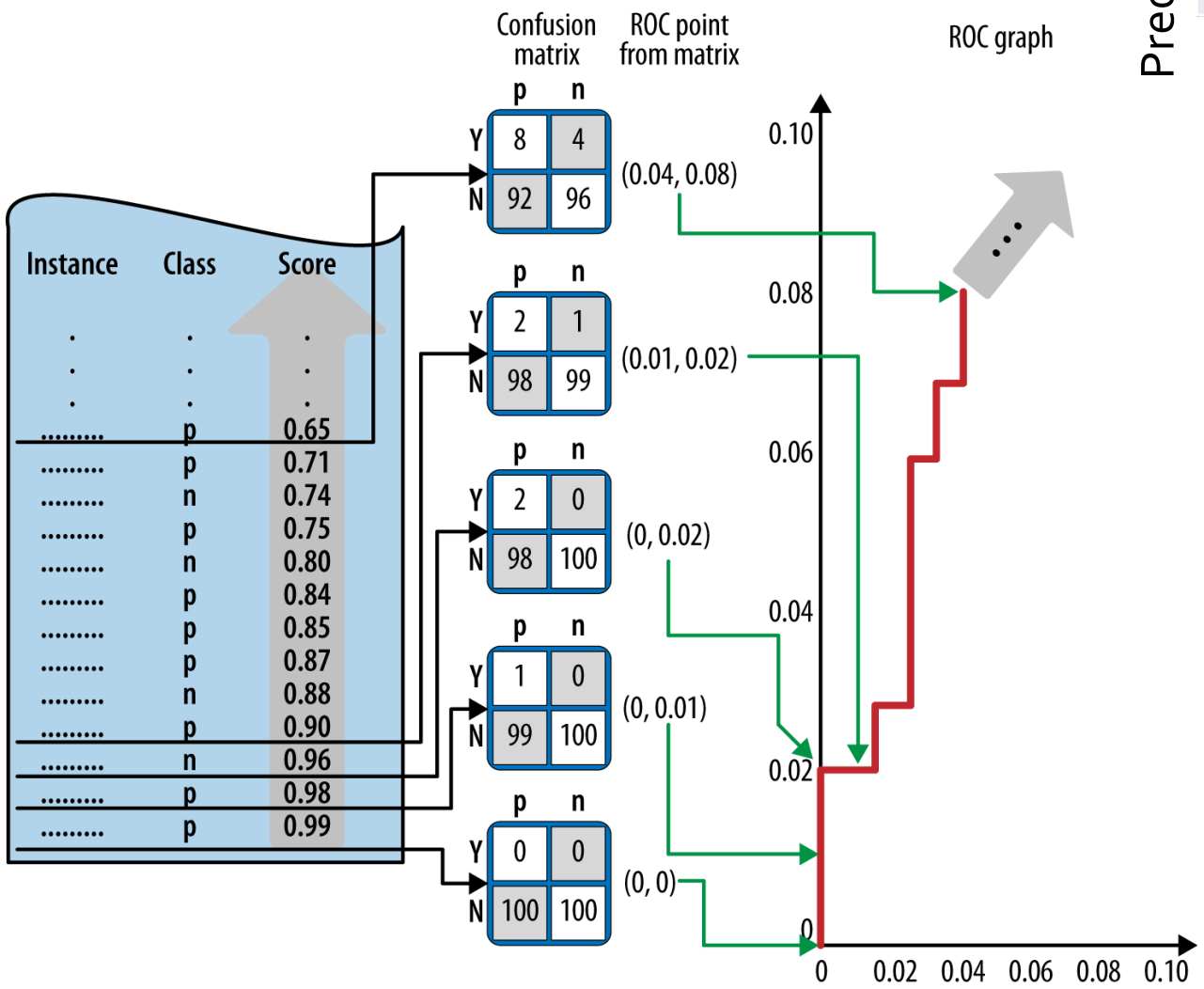


ROC Graphs and Curves (1)

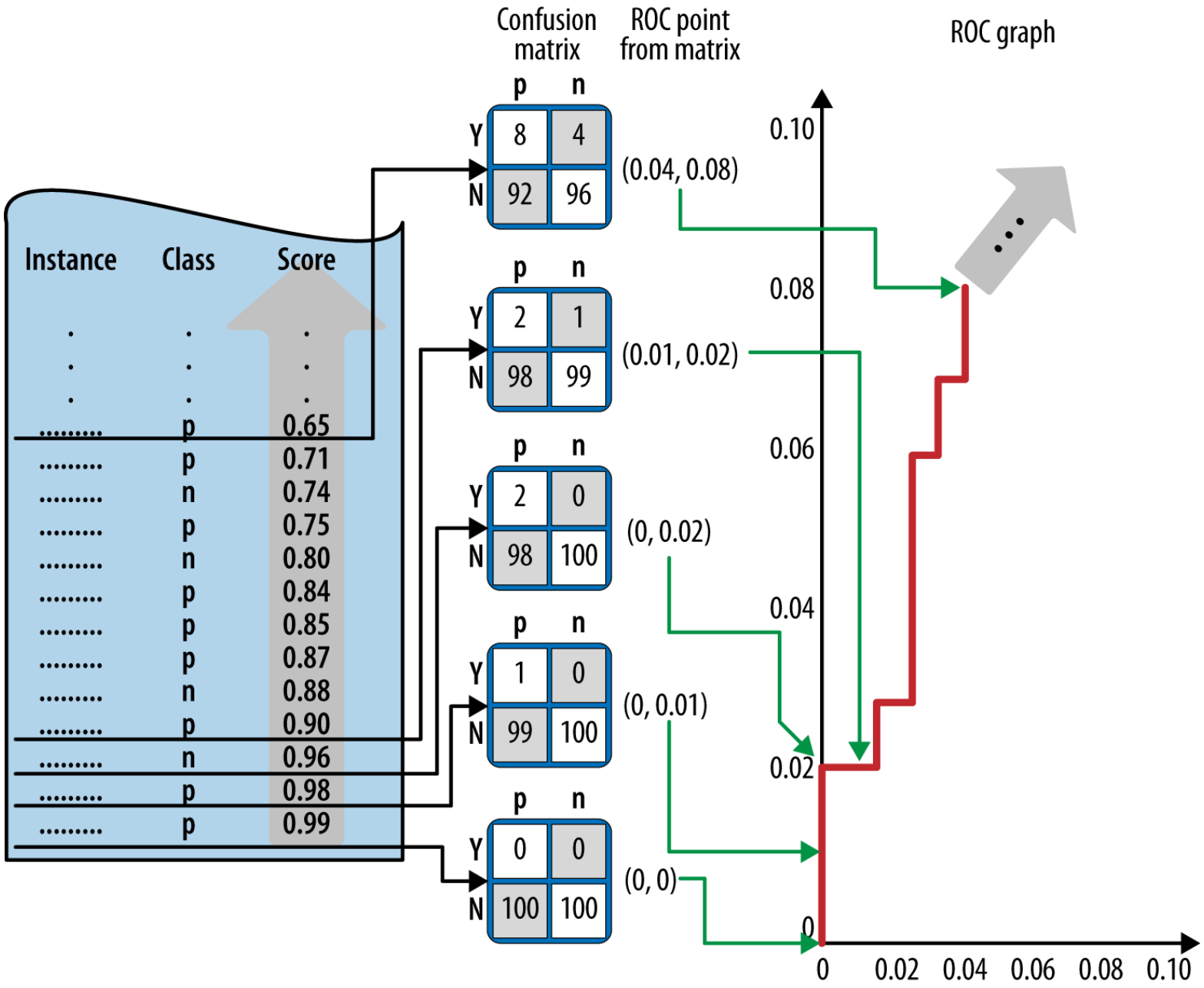
Actual/Real Class

Predicted Class	p	n
Y	True Positives	False Positives
N	False Negatives	True Negatives

Warning! This *confusion matrix* was **transposed** and different from the one before



ROC Graphs and Curves (2)



Generating ROC curve: Algorithm

- Sort the test set by the model predictions
- Start with cutoff = max (prediction)
- Decrease cutoff, after each step count the number of true positives TP (positives with prediction above the cutoff) and false positives FP (negatives above the cutoff)
- Calculate TP rate (TP/P) and FP (FP/N) rate
- Plot current number of TP/P as a function of current FP/N

ROC Graphs and Curves

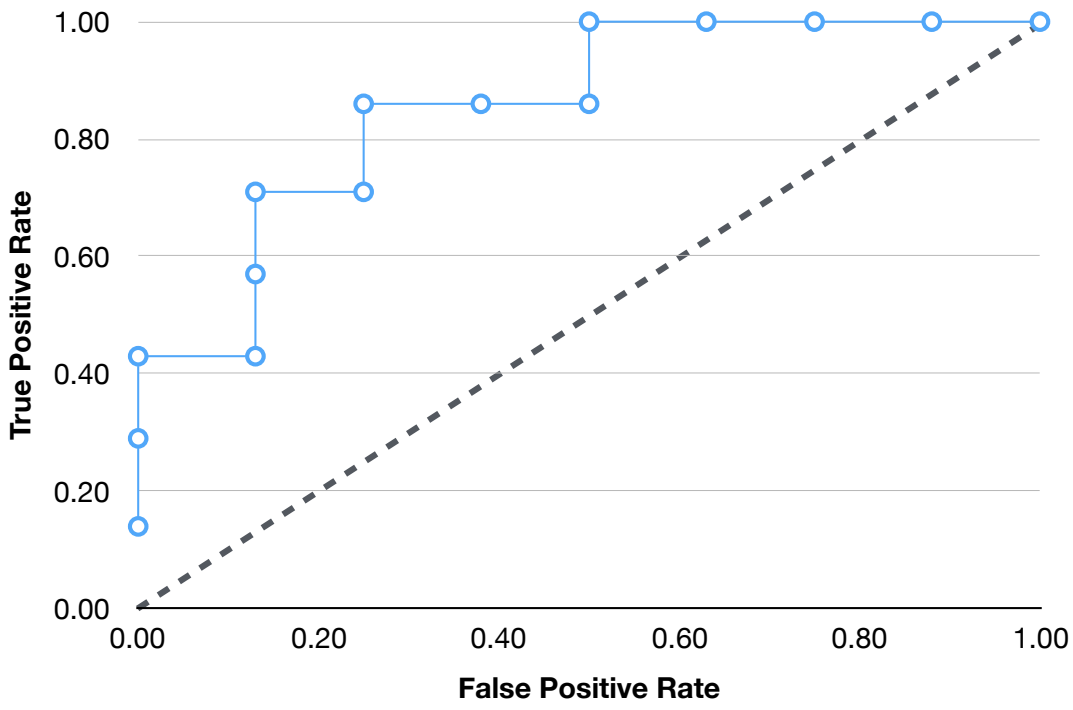
- ROC graphs decouple classifier performance from the conditions under which the classifiers will be used
- ROC graphs are independent of the class proportions as well as the costs and benefits
- Not the most intuitive visualization for many business stakeholders

Example: ROC Analysis

- Given a ranking classifier which will score test samples with a probability P of belonging to the positive class.
- Decision threshold θ controls whether a sample will be classified as positive or negative - i.e. $P > \theta$

Single example $\theta = 0.5$

P	> 0.5	Real
0.99	1	1
0.90	1	1
0.80	1	1
0.85	1	0
0.70	1	1
0.70	1	1
0.65	1	0
0.60	1	1
0.45	0	0
0.45	0	0
0.40	0	1
0.30	0	0
0.20	0	0
0.20	0	0
0.20	0	0

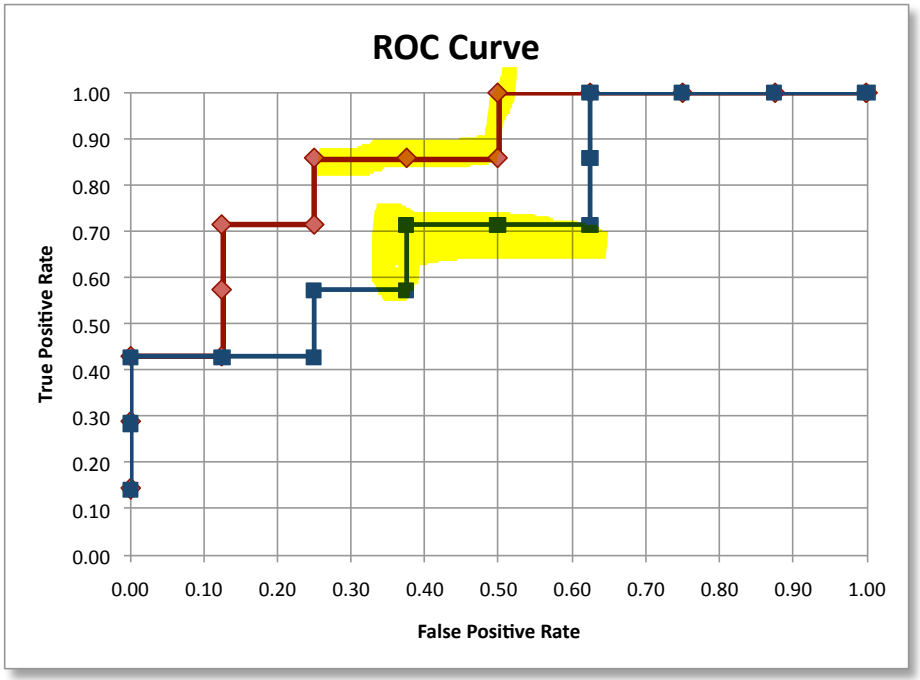


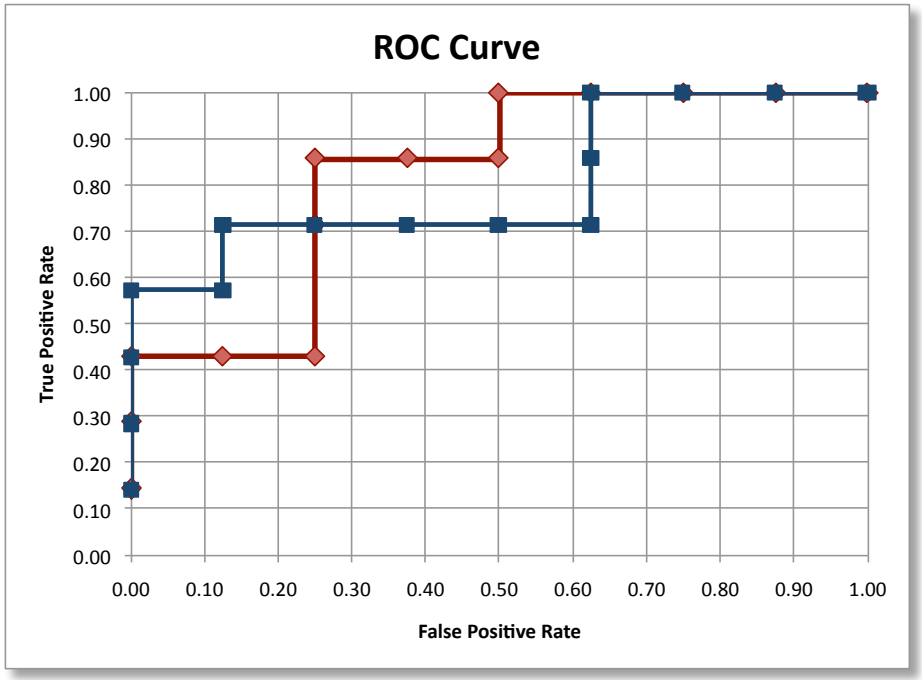
Visualizing Model Performance

- Profit Curves
- Receiver Operating Characteristic (ROC Curves)
- **Area Under the ROC Curve (AUC)**
- Cumulative Response curve and Lift curve

Comparing ROC Curves

- Often want to compare the performance of two classifiers at different thresholds → we can look at their ROC curves.
- In some cases, one classifier will always be better than another across all values of θ. In other cases, it will be more complicated…





➡ Make comparisons based on Area Under the Curve (AUC). A better classifier will have a ROC curve closer to top-left corner, giving a larger area under the curve.

Area Under the ROC Curve (AUC)

- The area under a classifier's curve expressed as a fraction of the unit square
  - Its value ranges from zero to one
- The AUC is useful when a single number is needed to summarize performance, or when nothing is known about the operating conditions
  - A ROC curve provides more information than its area
- Equivalent to the Mann-Whitney-Wilcoxon measure
  - Also equivalent to the Gini Coefficient (with a minor algebraic transformation)
  - Both are equivalent to the probability that a randomly chosen positive instance will be ranked ahead of a randomly chosen negative instance

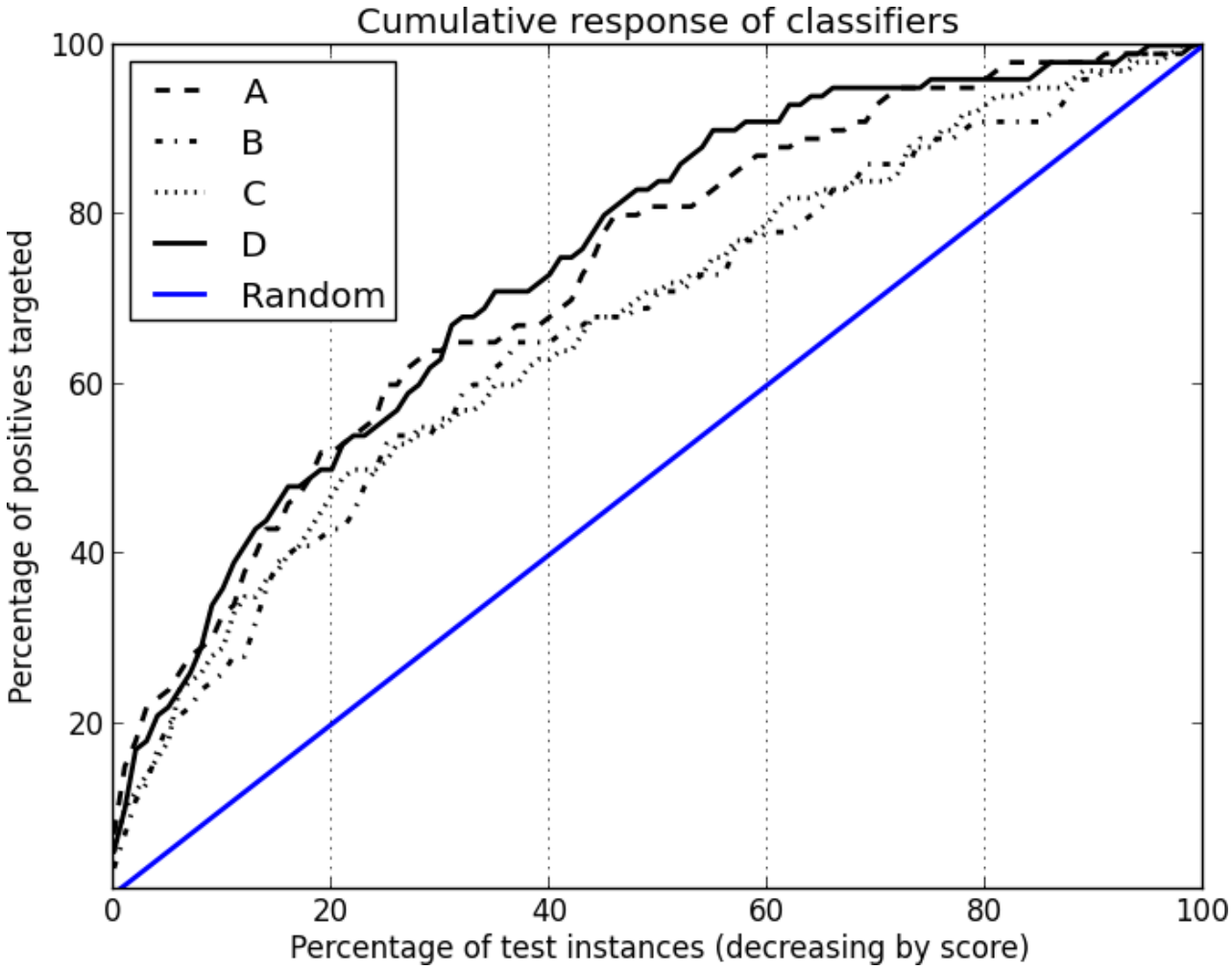
Visualizing Model Performance

- Profit Curves
- Receiver Operating Characteristic (ROC Curves)
- Area Under the ROC Curve (AUC)
- **Cumulative Response curve and Lift curve**

Cumulative Response curve

True Positive (TP) Rate / Hit tare

i.e. "the percentage of positives correctly classified"

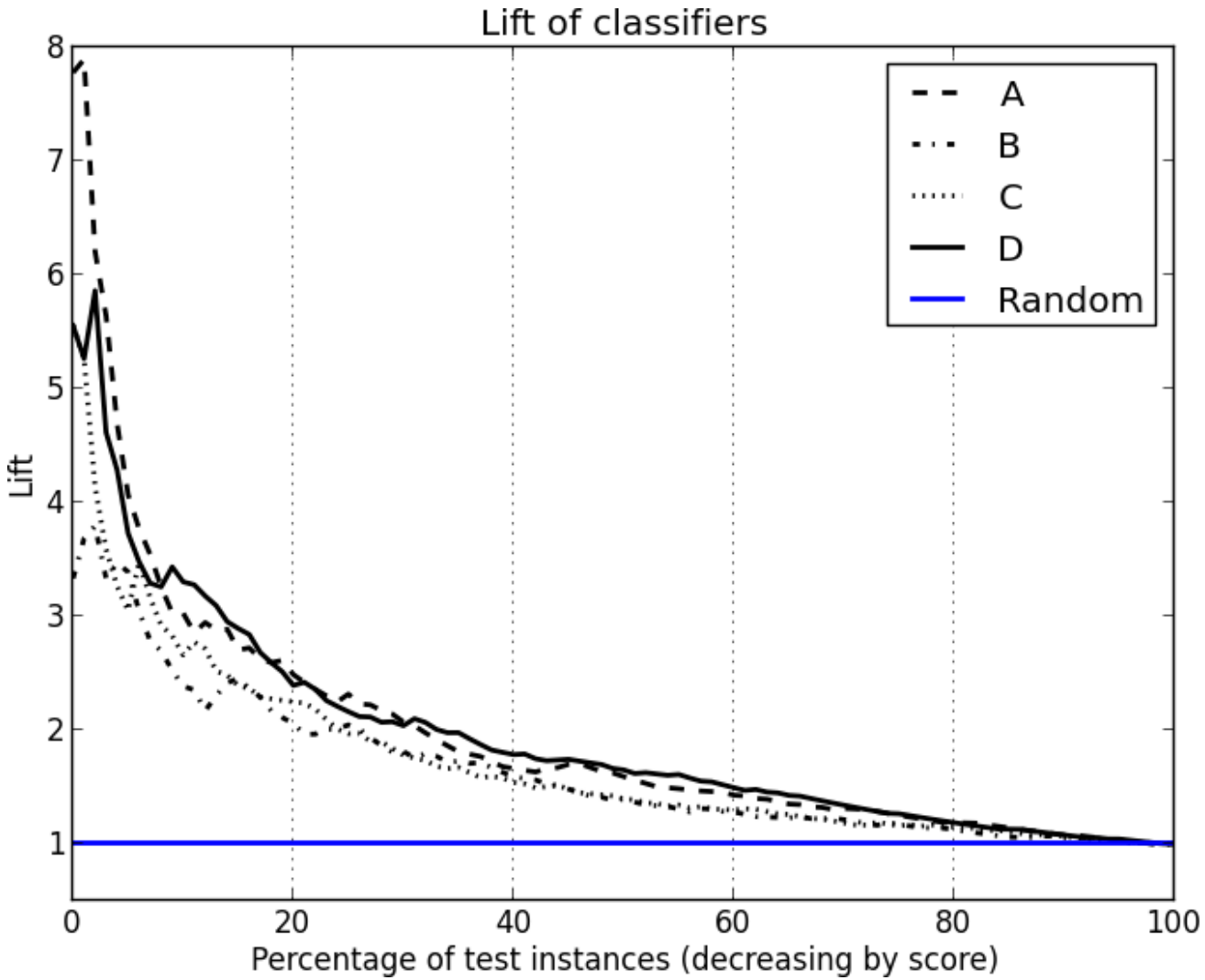


The percentage of the population is targeted

- Instances in a list are ordered by an effective ranking classifiers.
- At midway point (50) of 100 test instances (customers), who 50 churn and 50 do not.
  - Randomly sorted, giving a lift of 0.5/0.5 =1.
  - Perfect classifier, giving a lift of 1.0/0.5 = 2.

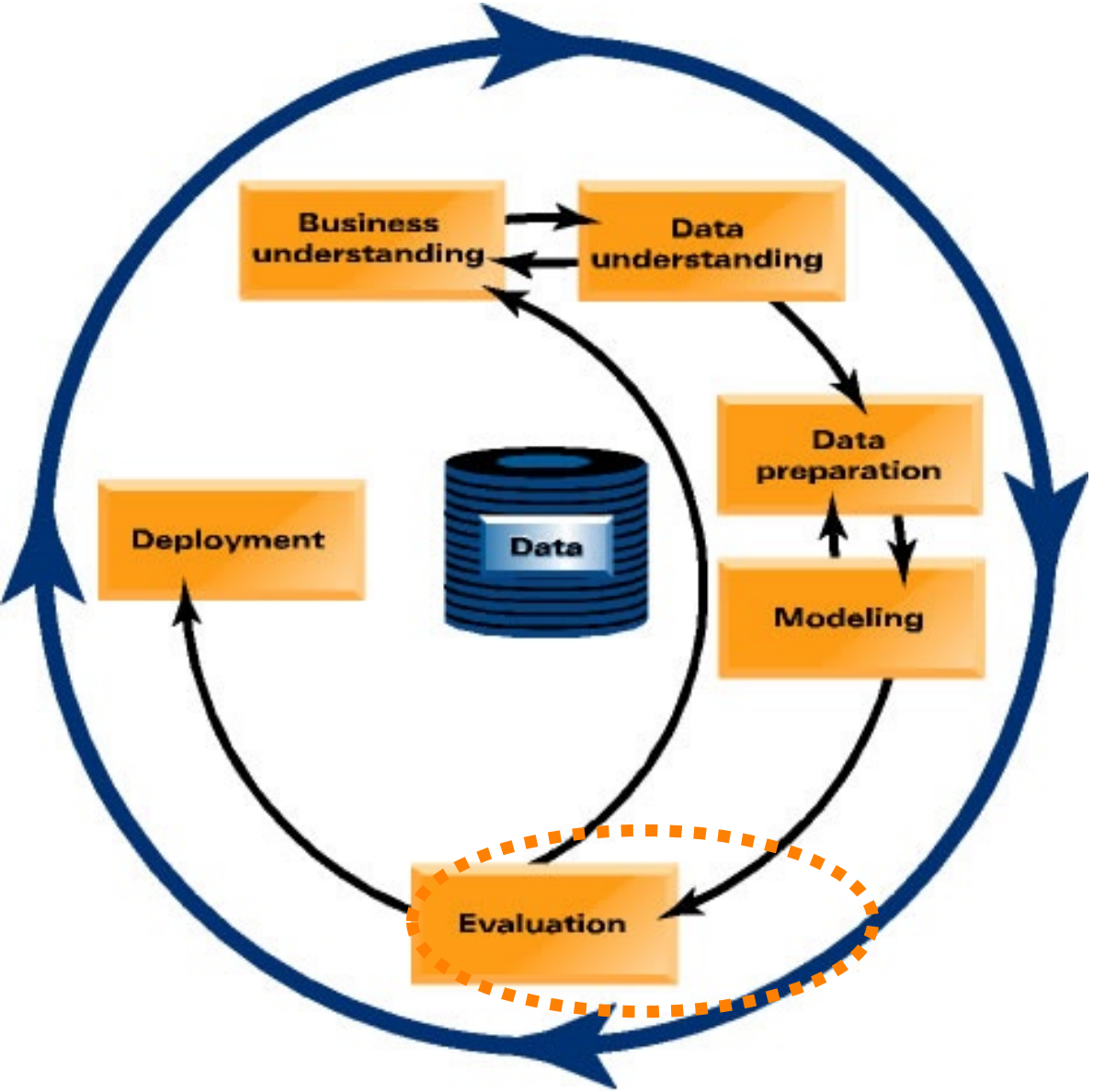
Lift Curve

The numeric lift



The percentage of the population is targeted

Let's focus back in on actually mining the data..



Which model should TelCo select in order to target customers with a special offer, prior to contract expiration?

Performance Evaluation

- Training Set:

Model	Accuracy
Classification Tree	95%
Logistic Regression	93%
k-Nearest Neighbors	100%
Naïve Bays	76%

- Test Set:

ยิ่งเยอะ ยิ่งดี

Model	Accuracy	AUC
Classification Tree	91.8%±0.0	0.614±0.014
Logistic Regression	93.0%±0.1	0.574±0.023
k-Nearest Neighbors	93.0%±0.0	0.537±0.015
Naïve Bays	76.5%±0.6	0.632±0.019

AUC เยอะแสดงว่าค่า threshold มันอ่อนไหวน้อย

Performance Evaluation

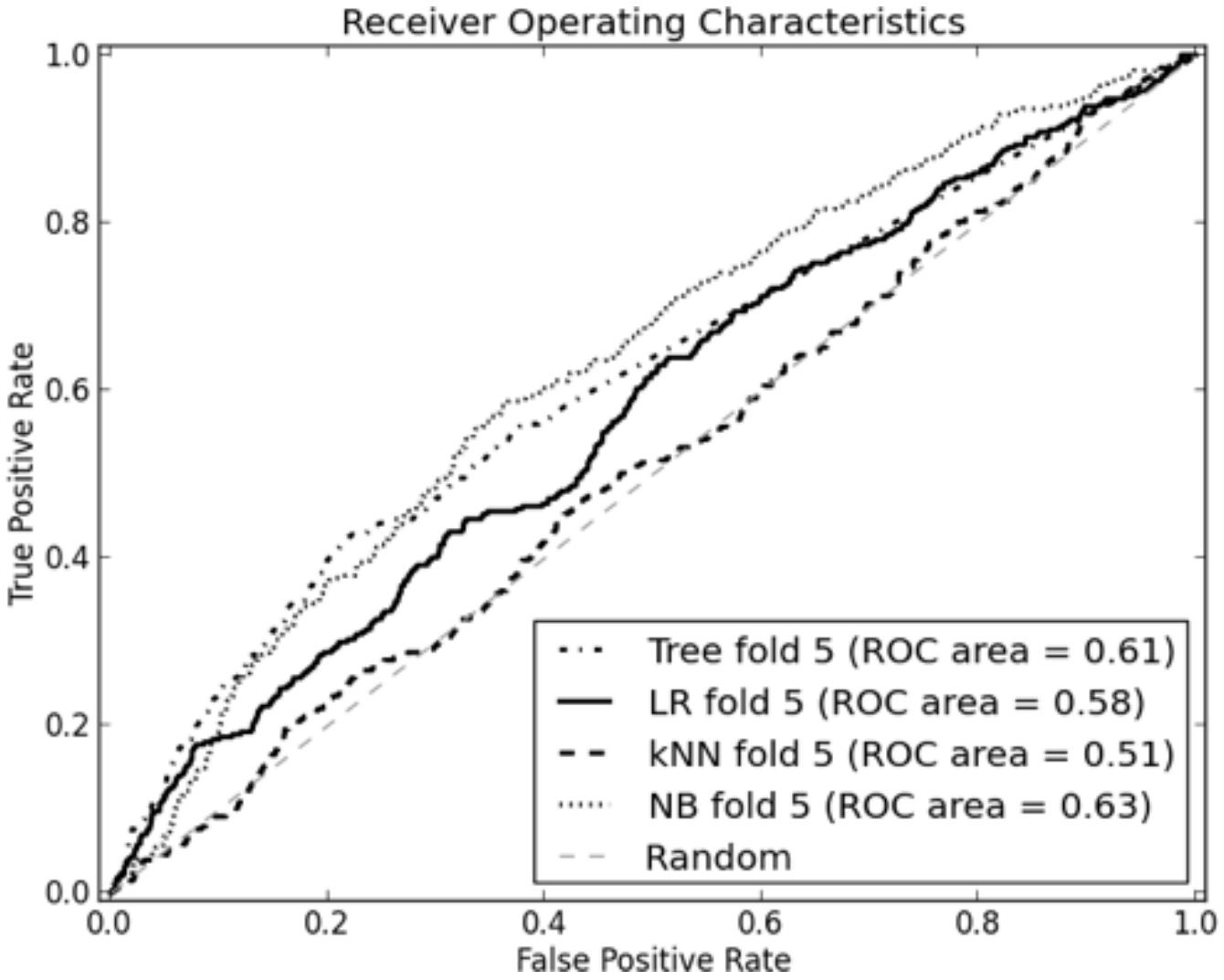
- Naïve Bayes confusion matrix:

	p	n
Y	127 (3%)	848 (18%)
N	200 (4%)	3518 (75%)

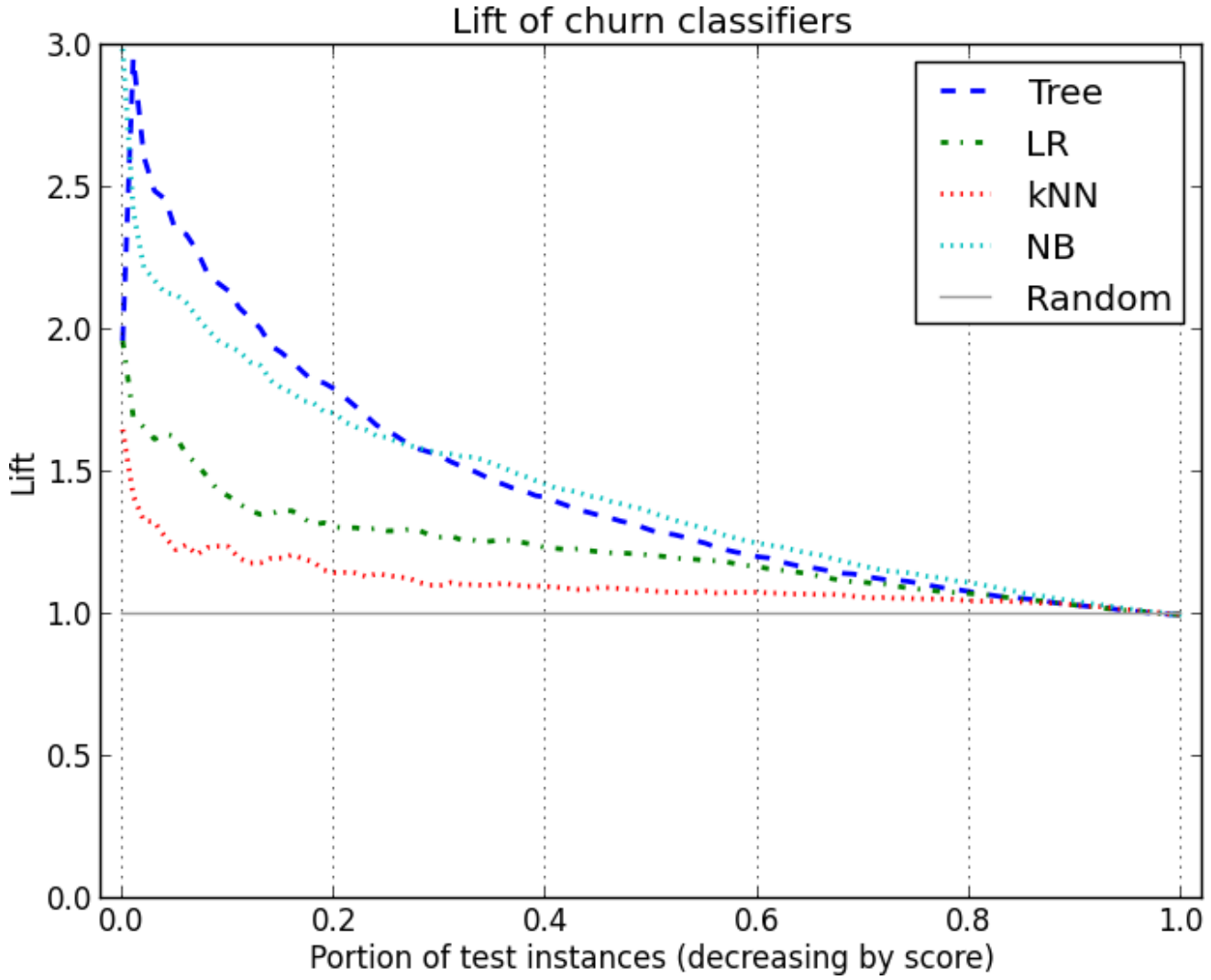
- k-Nearest Neighbors confusion matrix:

	p	n
Y	3 (0%)	15 (0%)
N	324 (7%)	4351 (93%)

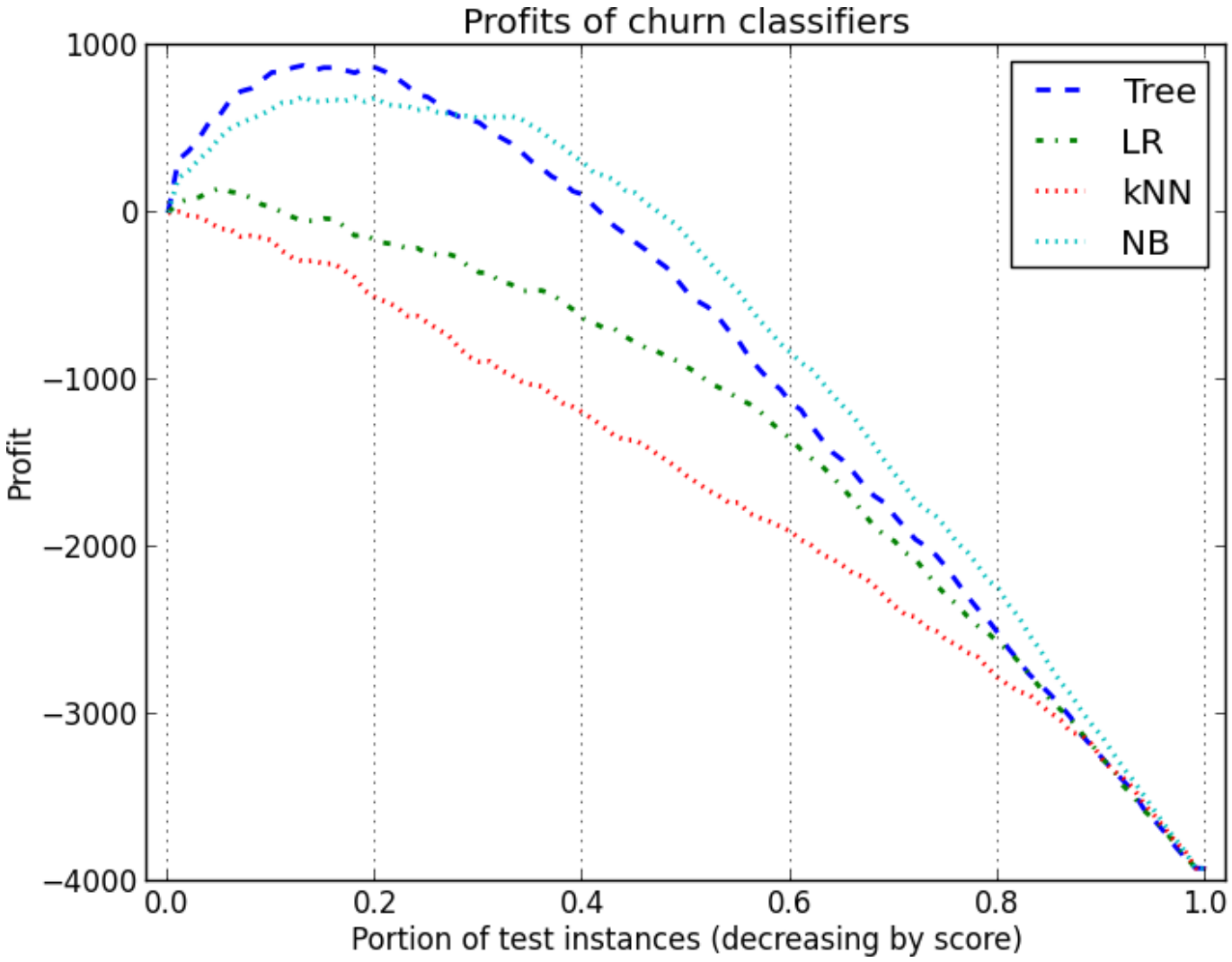
ROC Curve


Receiver Operating Characteristics
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Tree fold 5 (ROC area = 0.61)
LR fold 5 (ROC area = 0.58)
kNN fold 5 (ROC area = 0.51)
NB fold 5 (ROC area = 0.63)
Random

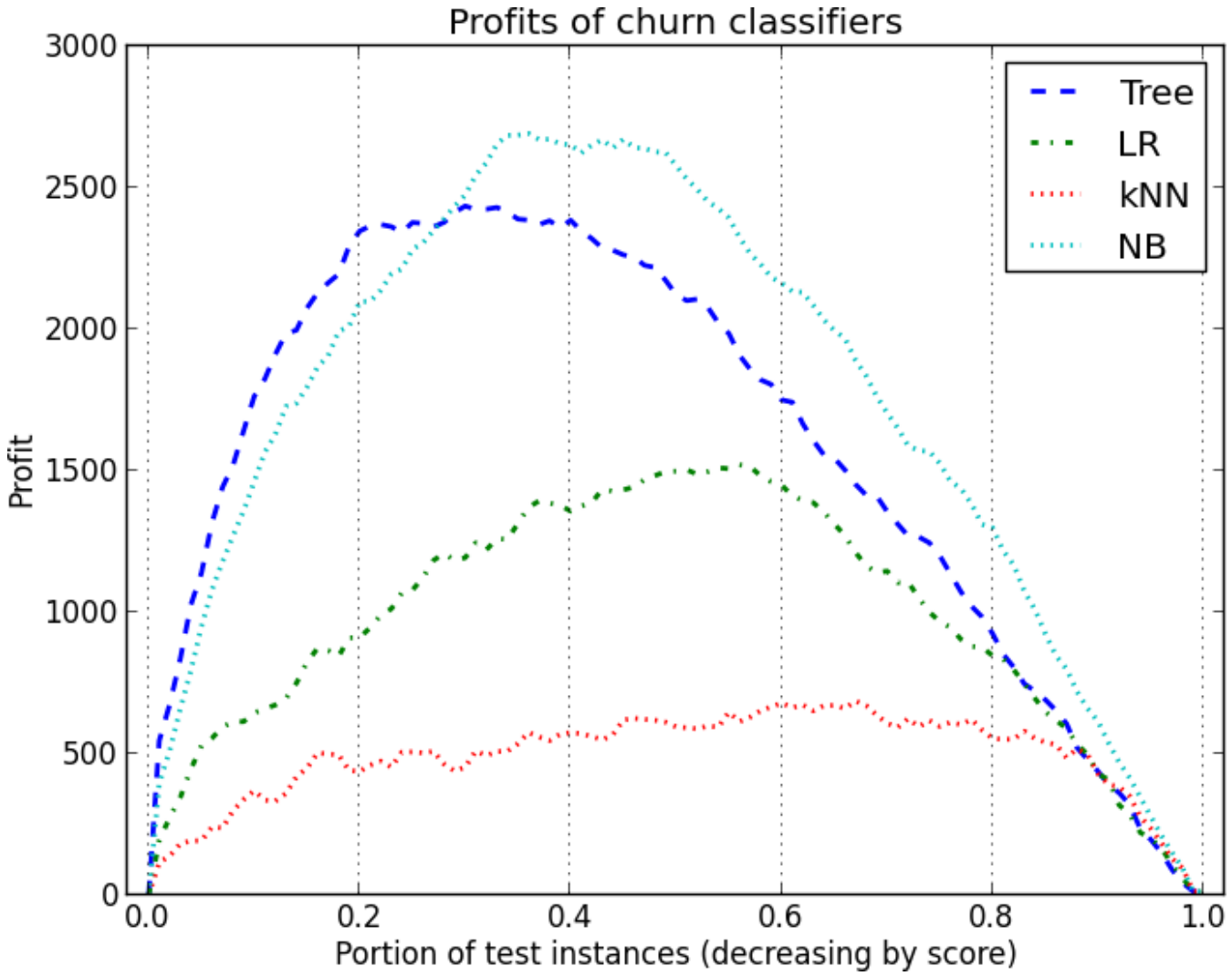
Lift Curve


Lift of churn classifiers
Tree
LR
kNN
NB
Random
Lift
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Portion of test instances (decreasing by score)

Profit Curves



Profit Curves



Data Science for Business
Experimental Setup

Asst. Prof. Teerapong Leelanupab (Ph.D.)
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang (KMITL)

Week
10.3

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.

Why de we need to evaluation?

Essentially, all models are wrong, but some are useful.

—George E. P. Box

- Allow systematic and objective comparison between different machine learning models in prediction.
- Most common *machine learning* tasks are:
  - Classification
  - Regression

Evaluation & Performance

```
from sklearn.metrics import accuracy_score, r2_score

accuracy_score(y_true, y_predict)  # Classification
mean_squared_error(y_true, y_predict) # Regression
r2_score(y_true, y_predict)  # Regression
```

Evaluation

```
from numba import jit

@jit(nopython=True)
def foo(n):
    f = 1
    for i in range(1, n+1):
        f = f*i
    return f
```

Improve Performance

Other packages
- Cython
- Dask

Key Idea in Evaluating Models

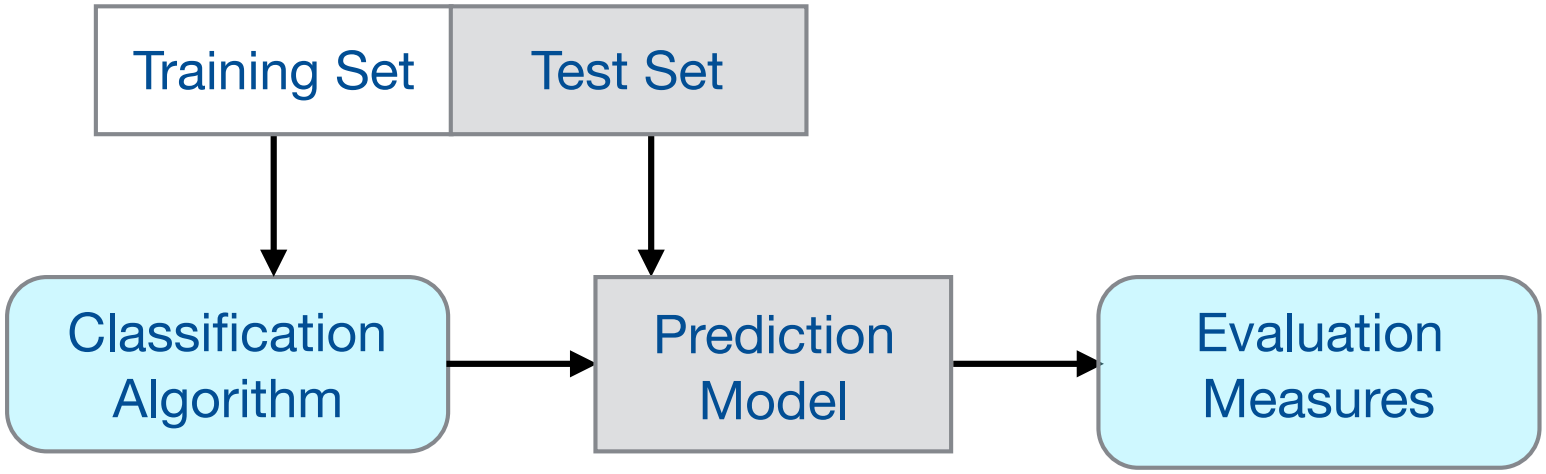
The most important rule in evaluating models is *not* to use the same data sample both to evaluate the performance of a predictive model and to train it.

We need to:

- divide our data into 2-3 sets, i.e.
  - training set and test set, or
  - training set, validation set and test set

Simple Hold-Out Strategy

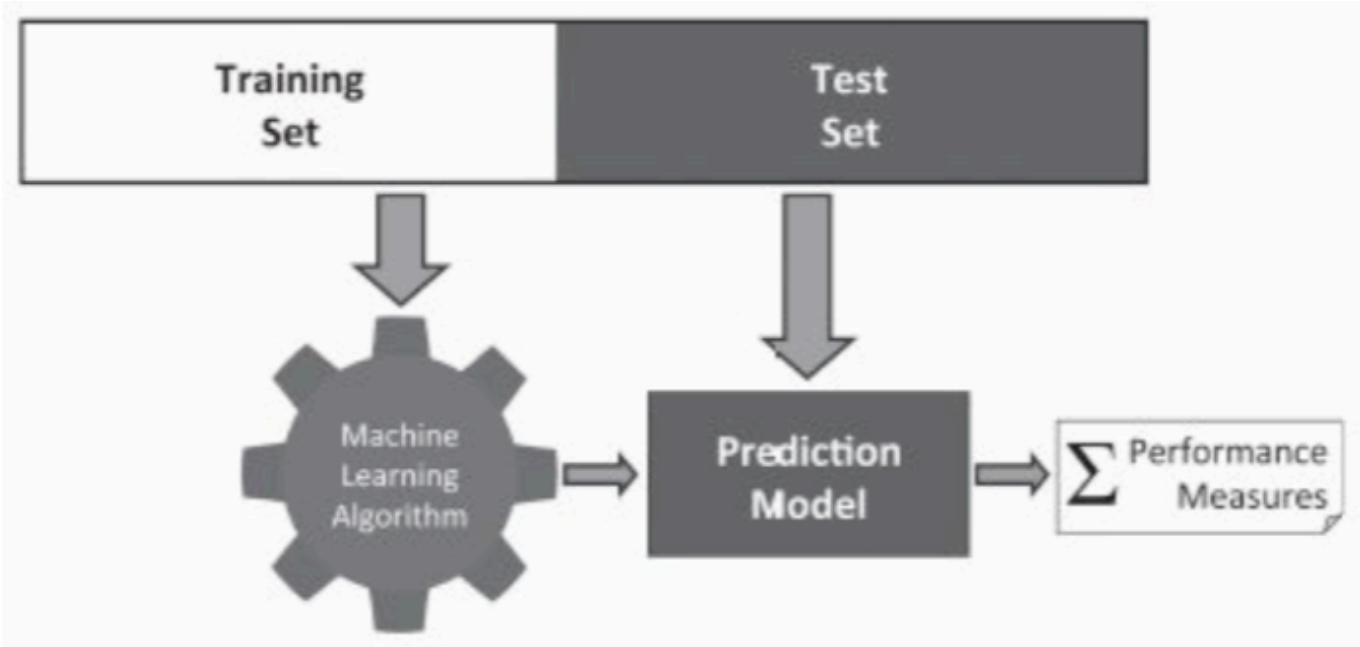
- Keep some training data back (the hold-out set) to use for evaluating the model produced by the classifier.
- Use performance on the hold-out set as a proxy for performance on unseen data (i.e. generalisation accuracy).



- Using a hold-out set avoids peeking - when the performance of a model is evaluated using the same data used to train it.
  e.g. Use of same training data for testing in Weka can produce unrealistic accuracy results that are “too good to be true”.

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

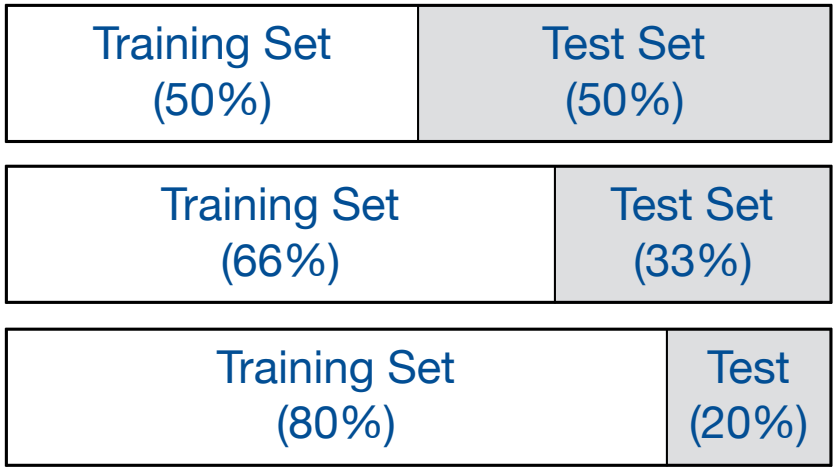
The simplest form of sampling



Take distinct, random, *non-overlapping* samples from an entire dataset and put them into training and test sets

Simple Hold-Out Strategy

- **Random Split:** Obtain a hold-out set by randomly assigning examples to either the training or test set with some probability.



! Sometimes we don't the “luxury” of setting aside data for testing.

! Since it is a single experiment, the hold-out estimate of error rate can be misleading if we get an “unfortunate” split of the data

! Even if we use multiple splits, some examples will never included for training or testing, while others might be selected many times.

Extension and Variations:

No single experimental design that is appropriate for all experimental scenarios

Classification

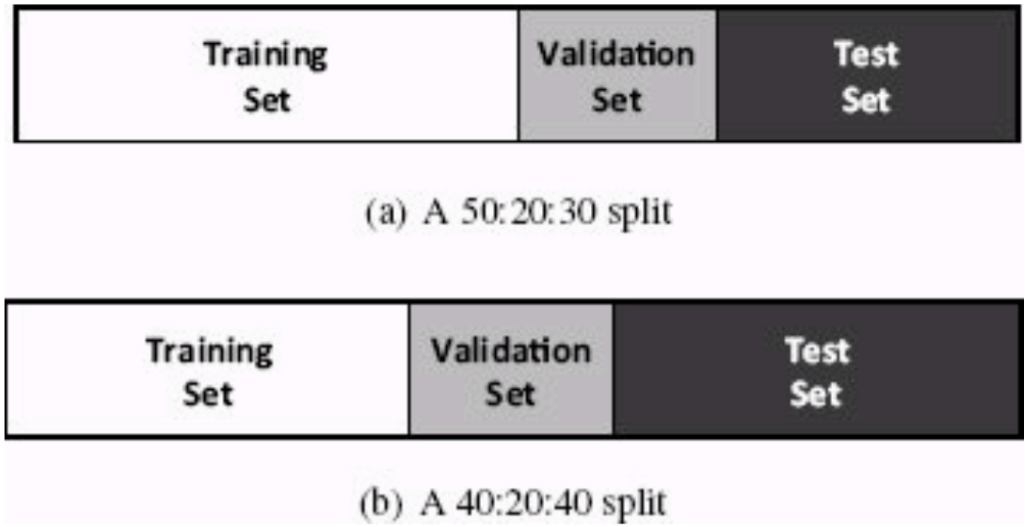
1) **Hold-out sampling with validation set**
2) ***k*-Fold Cross Validation**
3) **Leave-one-out Cross Validation (Jackknifing)**
4) **Bootstrapping**

Classification/Regression when a time dimension is concerned

1) **Out-of-time Sampling**
2) **Walk-forward Validation**

Hold-out Sampling, with Validation Set

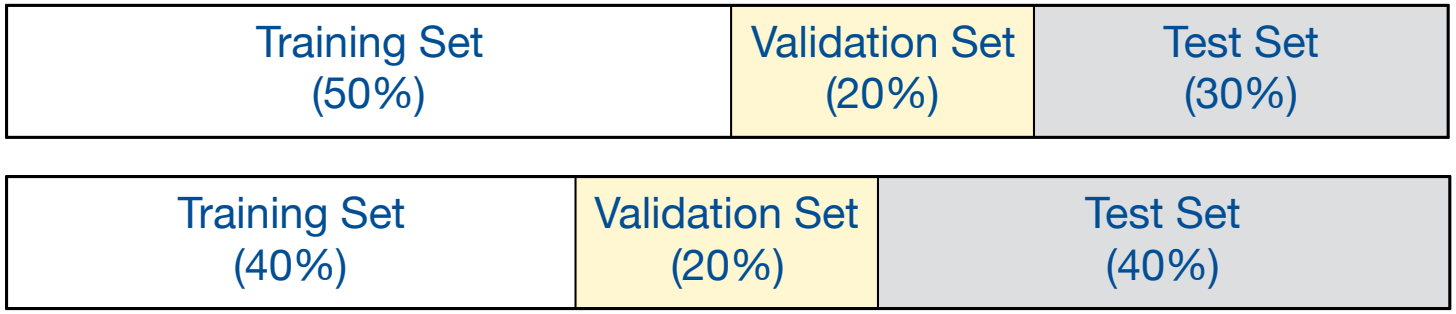
- Test set is ***not*** used in the process of training the model.
- Performance measured on this test set shows how well the model will perform on future *unseen data* after deployment
- Sometimes extended to include the third sample, the **validation set**



- No fixed recommendations for how to divide dataset into three different splits, although training:validation:test of 50:20:30 or 40:20:40 are common

Experimental Setup

- **Three-Way Hold-Out Strategy:** Divide the full dataset into three different subsets.
  1. Training set: The subset of examples used for learning.
  2. Validation set: The subset of examples used to tune the classifier (e.g. select parameter values).
  3. Test set: The subset of examples used only to assess the performance of a fully-trained classifier.



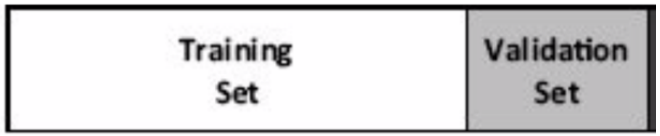
➡ This avoids a bias in evaluation of the model, where reusing examples from the validation set could lead to underestimates of the real error rate.

What is the validation set for?

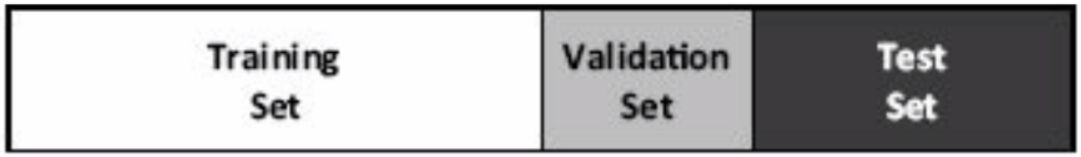
The validation set is used to tune particular aspects of a model, such as hyperparameters tuning, or how sensitive a model is as trained by different feature subset.

For example

First, train different models by varying their hyperparameters with 50% of dataset (training set) and test the trained models with 20% of dataset (validation set).



Second, train the best model with tuned hyperparameters with 70% of dataset (training set + validation set) and test the trained model with 30% of dataset (test set) to evaluate the expected performance of the model on future unseen data after deployment.



(a) A 50:20:30 split

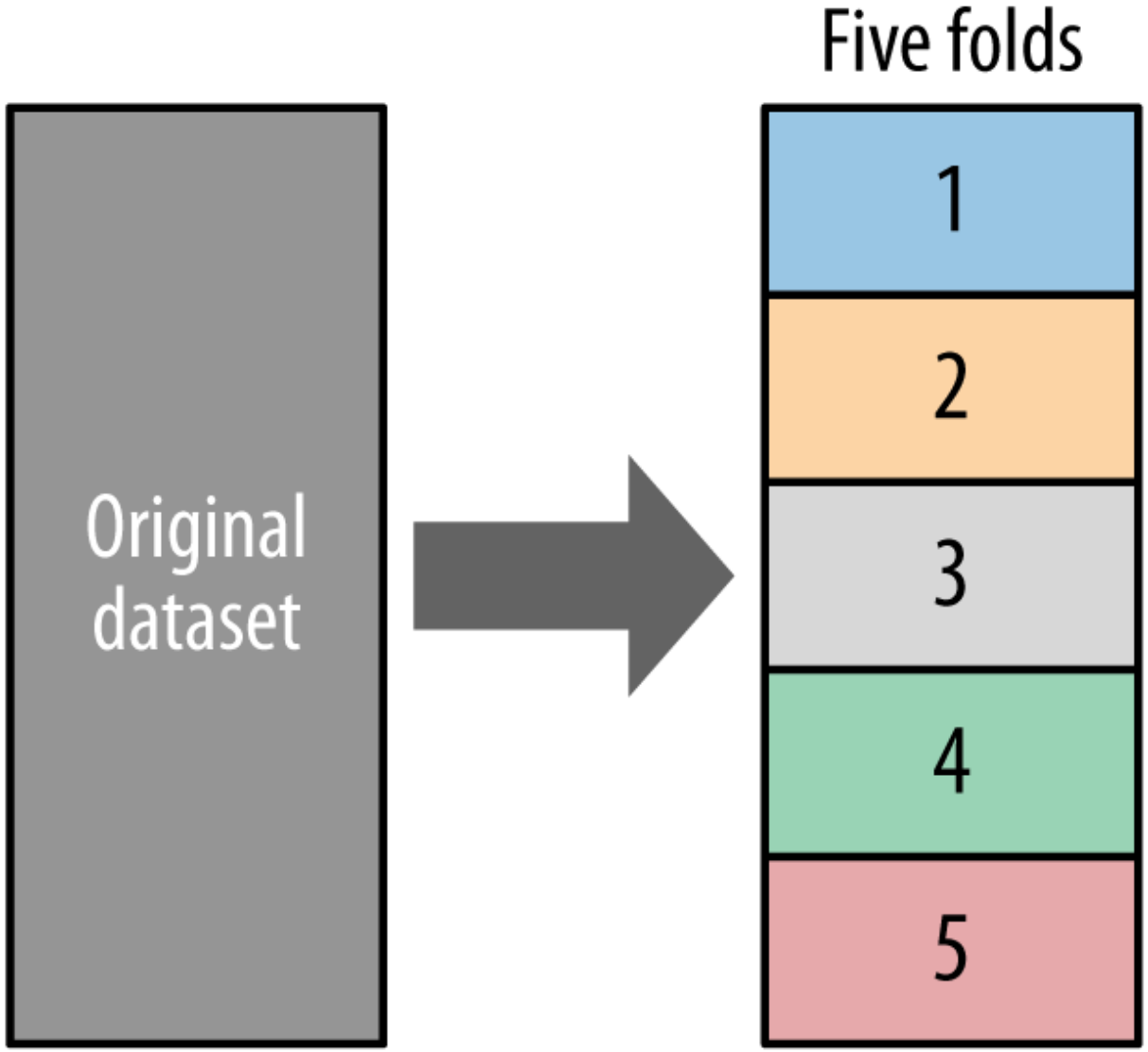
Problem of Hold-out Sampling

Lucky Splitting, placing the *difficult* instances into the training set and the *easy* ones in the test set. The created model will appear to be more accurate than it will actually be when deployed.

Cross-Validation

- The main purpose of Cross-Validation (CV) is to get an estimate of the generalization performance of the model.
- CV is **not intended to get one or multiple trained models for inference**, but to **estimate an unbiased generalization performance**.

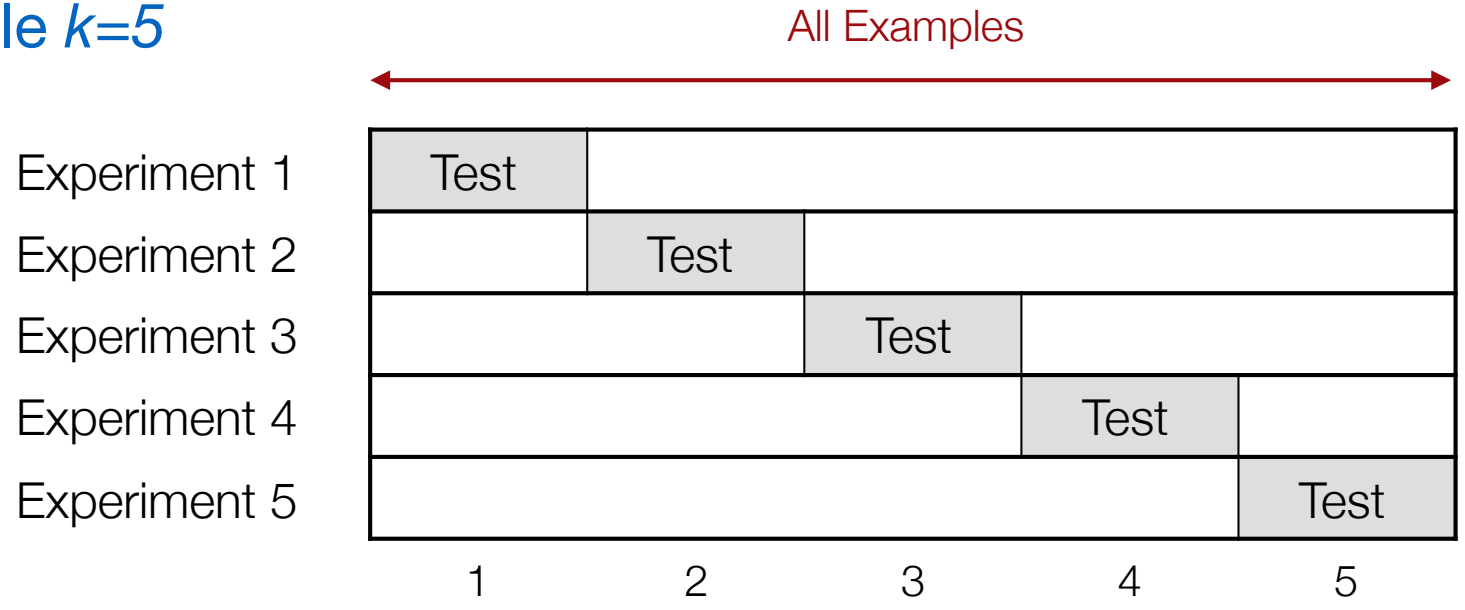
Cross-Validation



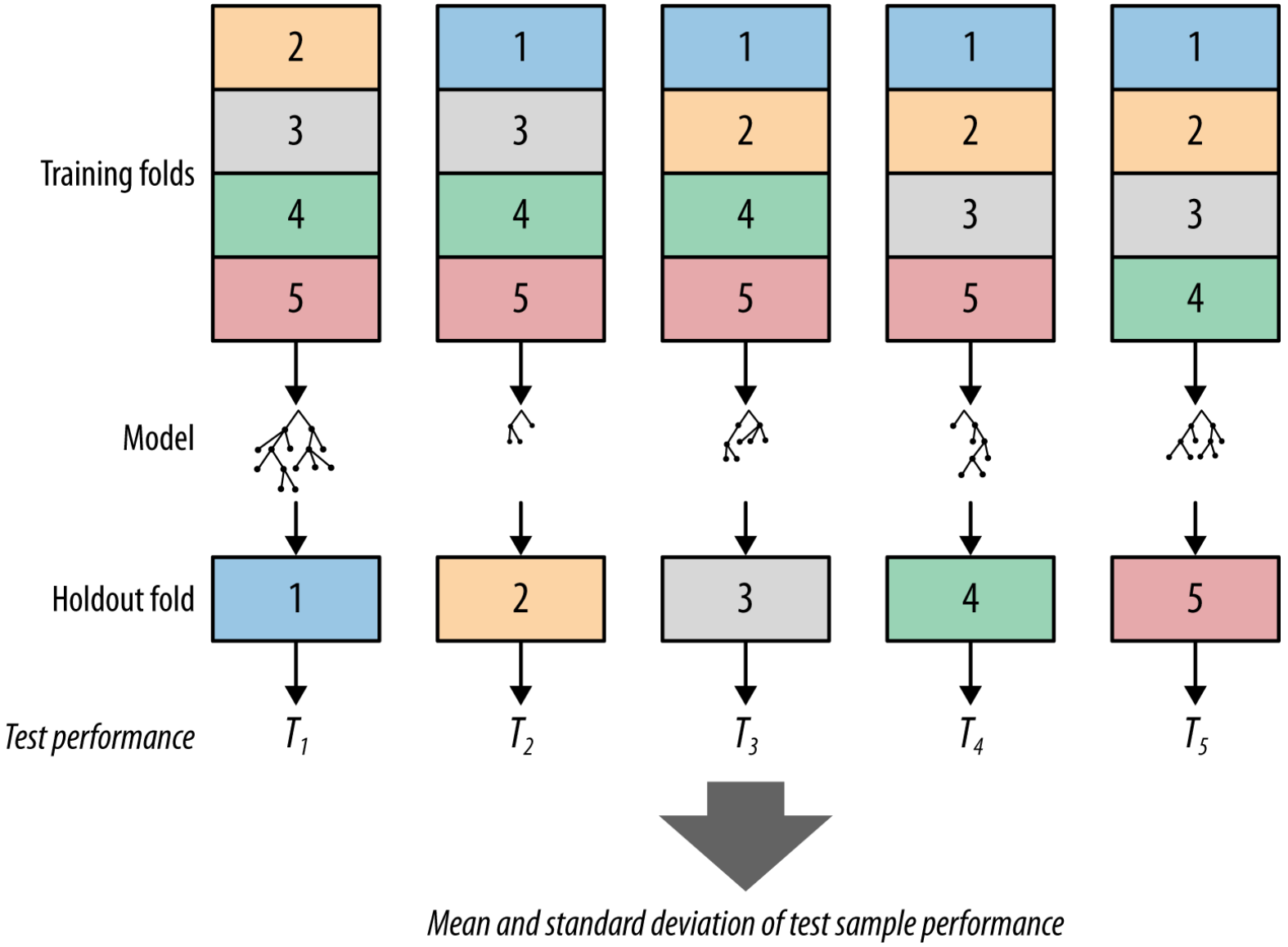
Cross Validation

- ***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*



Cross-Validation



K-fold cross validation (ex: *k*=10)

Dataset is divided into *k* equal-sized folds or partition, and *k* separate evaluation experiments are performed.

Training set

Test set

Report the results of *k*-fold validation, ex: *k* = 5 (1)

Fold	Confusion Matrix				Classification Accuracy
1			Prediction		81%
			lateral	*frontal*	
	Target	*lateral*	43	9	
		frontal	10	38	
2			Prediction		88%
			lateral	*frontal*	
	Target	*lateral*	46	9	
		frontal	3	42	
3			Prediction		82%
			lateral	*frontal*	
	Target	*lateral*	51	10	
		frontal	8	31	

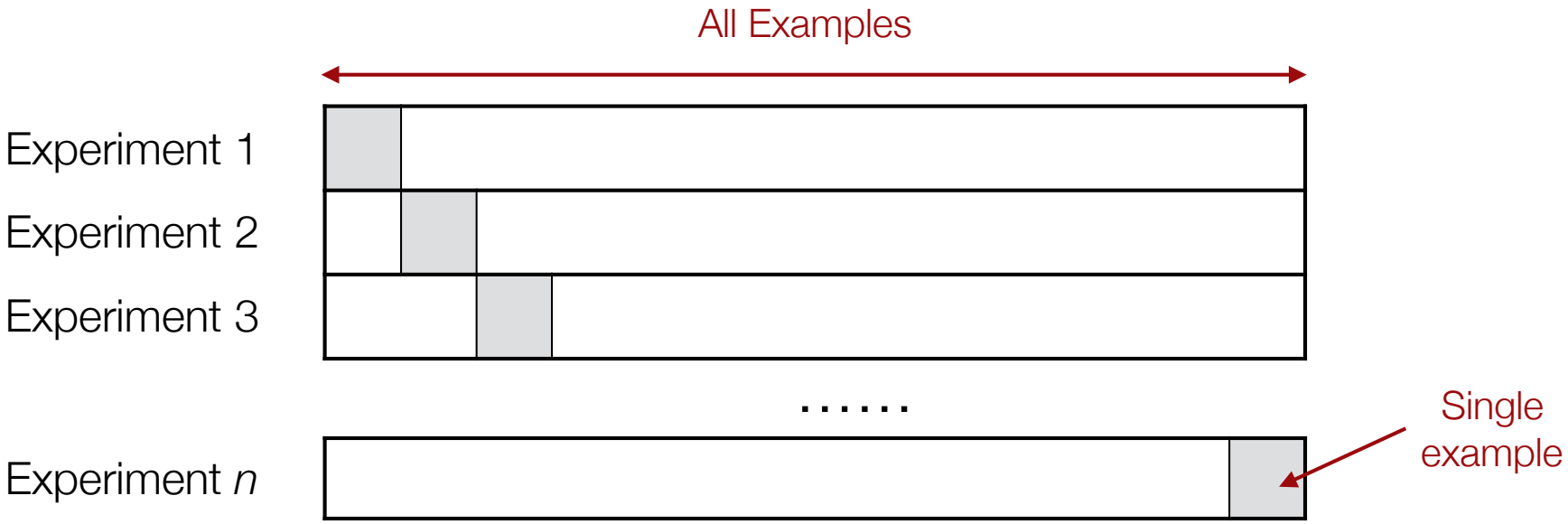
Report the results of *k*-fold validation, ex: *k* = 5 (2)

Fold	Confusion Matrix				Classification Accuracy
4			Prediction		85%
			lateral	*frontal*	
	Target	*lateral*	51	8	
		frontal	7	34	
5			Prediction		84%
			lateral	*frontal*	
	Target	*lateral*	46	9	
		frontal	7	38	
Overall			Prediction		84%
			lateral	*frontal*	
	Target	*lateral*	237	45	
		frontal	35	183	

After estimating the performance of a model using *k*-fold cross validation, with parameters set according to the validation, we typically train a model that will be deployed using *all* of available data.

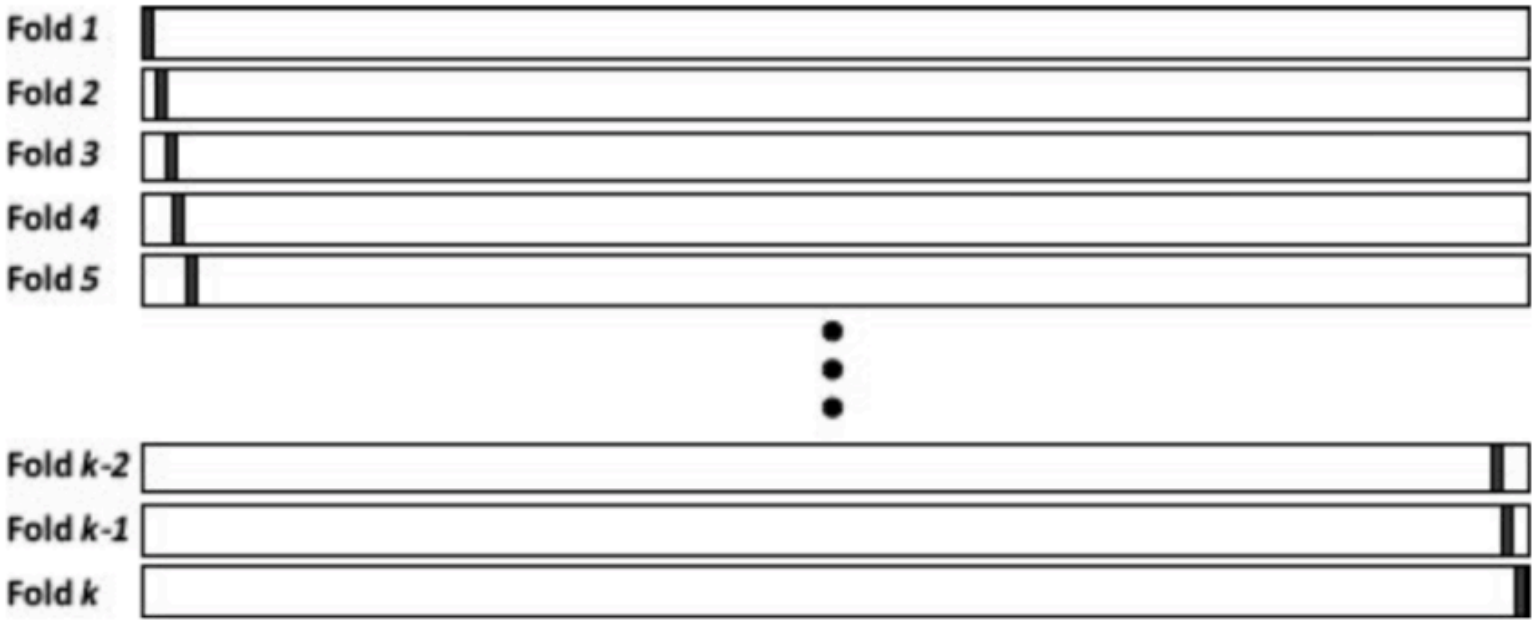
Leave-one-out Cross Validation

- **Leave-one-out:** 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.



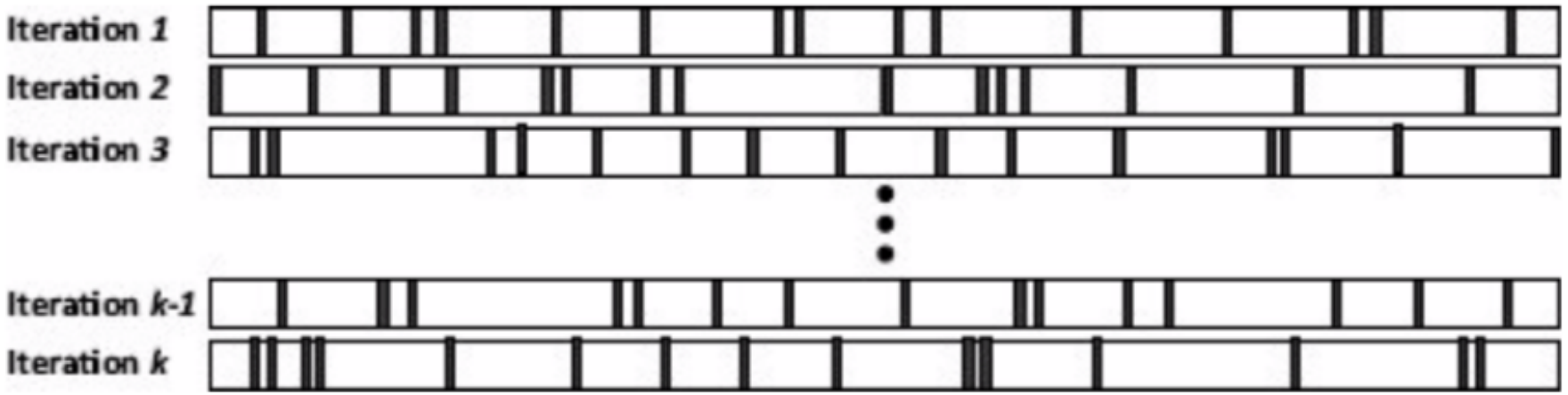
Leave-one-out Cross Validation

- Extreme form of *k*-fold cross validation, where *k* = the number of all instances
- Suitable when dataset is too small to allow big enough training sets in a *k*-fold cross validation
- In each fold, the test set contains only one instance.
- The training set contains the remainder of the data.



Bootstrapping

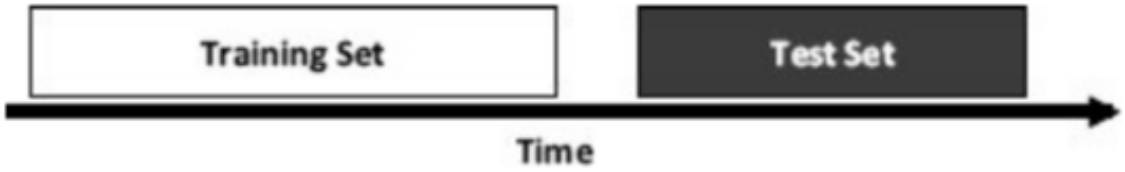
- Suitable when dataset is even very smaller than those used in leave-one-out cross validation (fewer than 300 instances)
- In each iteration (typically *k* is over 200 iterations), perform randomly sampling (bootstrap with/without replacement) to generate training and test sets
- *m* is the number of instances in a test set and the remainder is the number of instances in a training set.
- Note that some instances may be used more than once in test sets of different iterations.

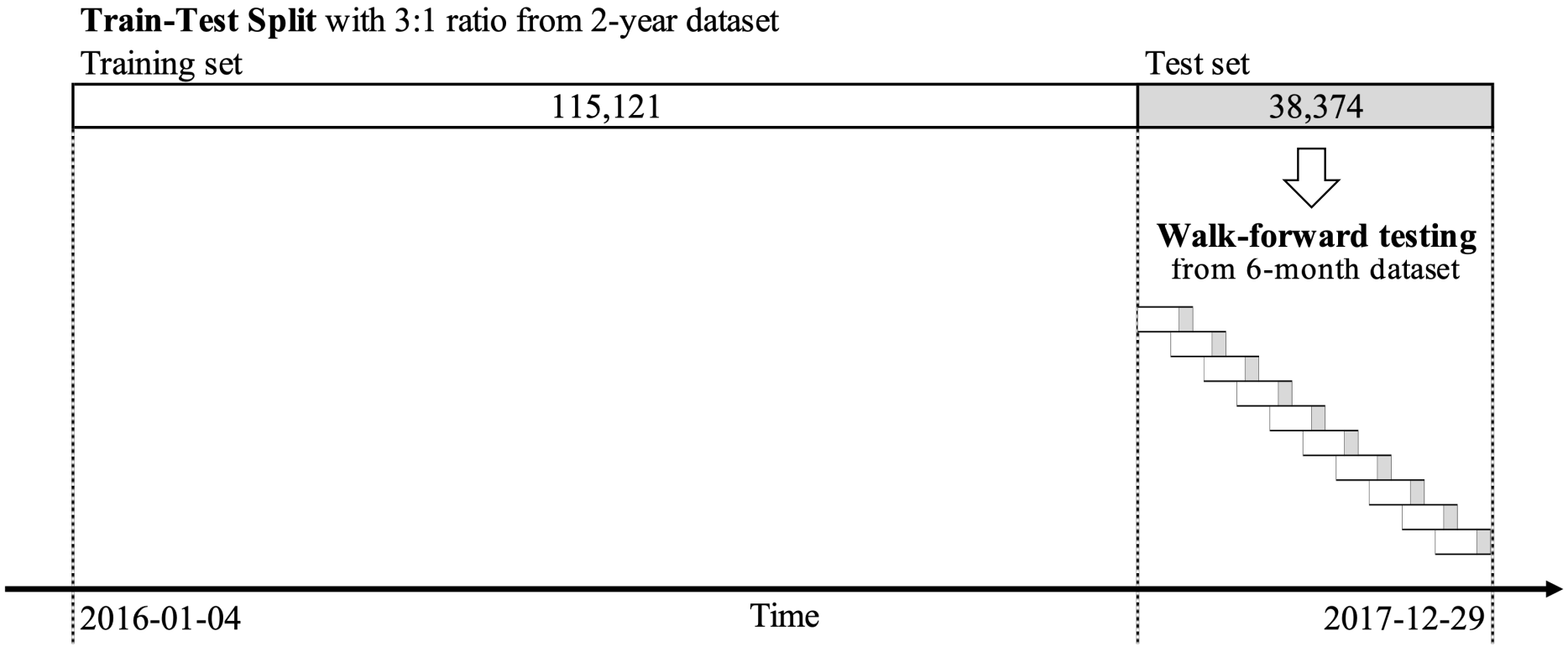


Simple Train-Test Split in Time Series

aka. Out-of-time validation

Consider a time dimension in splitting data

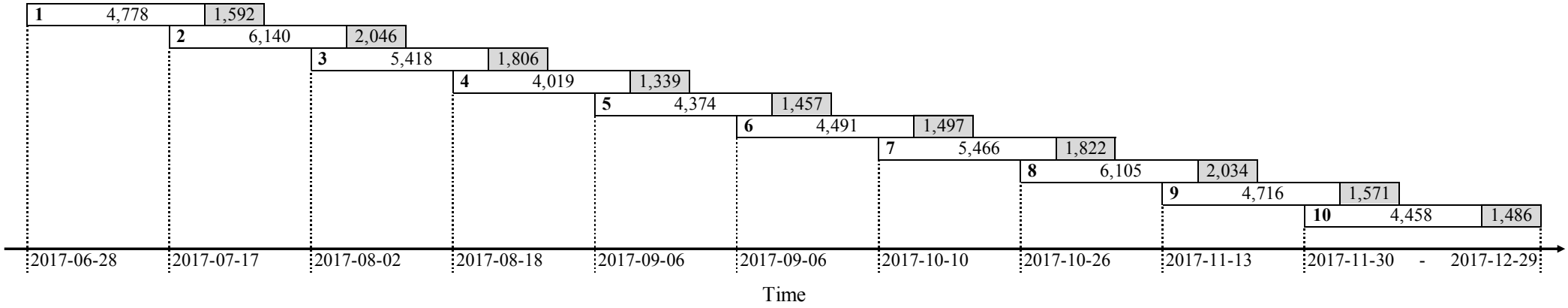




Walk-Forward Testing Routine

aka. Walk-Forward validation

Similar to *k*-fold cross validation, but need to consider about temporal aspect (ex. $k = 10$)



Summary

- Measuring Performance
  - Misclassification Rate & Accuracy
  - Precision & Recall
  - Balance accuracy measures
- ROC Analysis
- Overfitting
- Experimental Setup
  - Hold-out validation
  - *k*-Fold cross validation

References

- J. D. Kelleher, B. Mac Namee, A. D'Arcy. "Fundamentals of Machine Learning for Predictive Data Analytics", 2015.
- C. D. Manning, P. Raghavan and H. Schütze. "Introduction to Information Retrieval", Cambridge University Press. 2008
- E. Alpaydin. "Introduction to Machine Learning", Adaptive Computation and Machine Learning series, MIT press, 2009.
- J. Davis, M. Goadrich. "The relationship between Precision-Recall and ROC curves". Proceedings of ICML 2006.