

# Data Mining: Model Evaluation

Tom Heskes

# Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

# Model Evaluation

- **Metrics for Performance Evaluation**
  - How to evaluate the performance of a model?
- **Methods for Performance Evaluation**
  - How to obtain reliable estimates?
- **Methods for Model Comparison**
  - How to compare the relative performance among competing models?

# Metrics for Performance Evaluation (1)

- Focus on the predictive capability of a model
  - Rather than how fast it takes to classify or build models, scalability, etc.
- **Confusion Matrix:**

ACTUAL CLASS	PREDICTED CLASS	
	Class=Yes	Class=No
	Class=Yes	Class=No
	Class=Yes	Class=No
	TP	FN
	FP	TN

TP: true positive  
FN: false negative  
FP: false positive  
TN: true negative

- **FP:** falsely predicted to be **positive** (thus actual class is negative)
- **FN:** falsely predicted to be **negative** (thus actual class is positive)

## Metrics for Performance Evaluation (2)

ACTUAL CLASS	PREDICTED CLASS		
		Class=Yes	Class=No
	Class=Yes	TP	FN
	Class=No	FP	TN

- Most widely-used metric:

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

## Limitation of Accuracy

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10
- If model predicts everything to be class 0, the accuracy is  $9990/10000 = 99.9\%$ 
  - Accuracy is misleading because model does not detect any class 1 example
  - Should always at least compare with baseline accuracy!

## Cost Matrix

ACTUAL CLASS	PREDICTED CLASS		
		Class=Yes	Class=No
	Class=Yes	$C(\text{Yes} \text{Yes})$	$C(\text{No} \text{Yes})$
	Class=No	$C(\text{Yes} \text{No})$	$C(\text{No} \text{No})$

- $C(i|j)$ : Cost of misclassifying class  $j$  example as class  $i$
- Cost of misclassifying a healthy subject as a patient typically lower than vice versa

# Computing Cost of Classification

Cost Matrix	PREDICTED CLASS		
ACTUAL CLASS	$C(i j)$	+	-
	+	0	100
	-	2	0

Model $M_1$	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	90	10
	-	90	210

Accuracy = 75%

Cost = 1180

Model $M_2$	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	70	30
	-	50	250

Accuracy = 80%

Cost = 3100



## Cost-Sensitive Measures

- **Precision**: of all samples predicted to be positive predictions, what fraction is actually positive? Biased towards  $C(\text{Yes}|\text{Yes})$  &  $C(\text{Yes}|\text{No})$

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

- **Recall**: of all actually positive samples, what fraction is predicted to be positive? Biased towards  $C(\text{Yes}|\text{Yes})$  &  $C(\text{No}|\text{Yes})$

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

- **F-measure** balances Precision and Recall

$$\text{F - measure (F)} = \frac{2rp}{r + p} = \frac{2TP}{2TP + FP + FN}$$

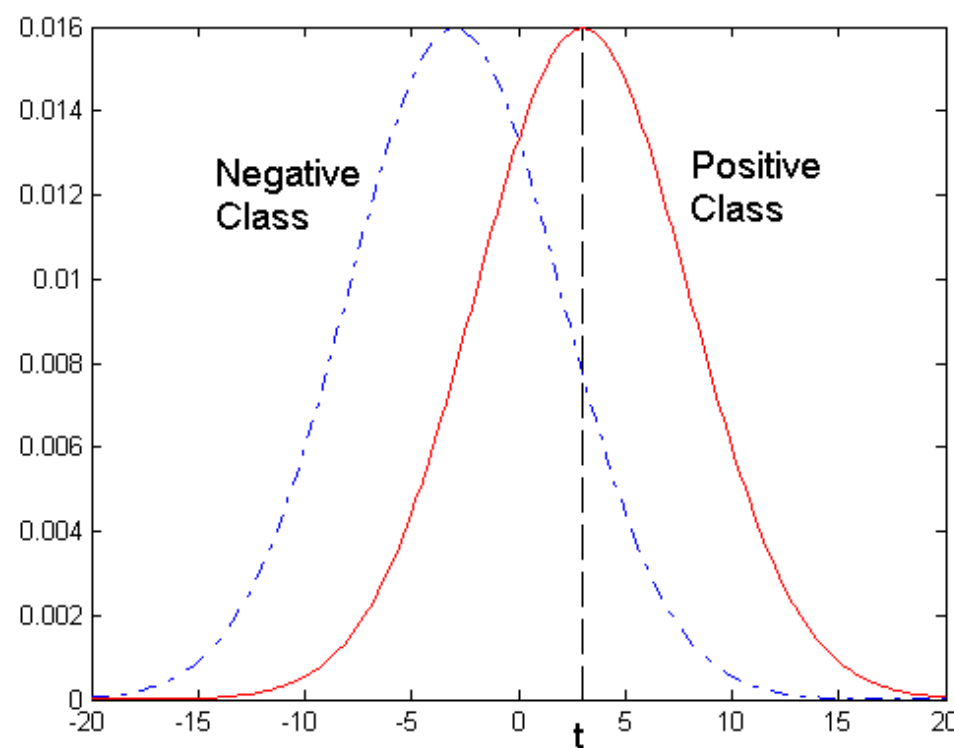
# Receiver Operating Characteristics (ROC)

- Developed in 1950s for signal detection theory to analyze noisy signals
  - Characterizes the trade-off between positive hits and false alarms
- ROC curve plots TP rate (on the y-axis) against FP rate (on the x-axis)
- TP rate =  $TP / (TP + FN)$  = recall = sensitivity  
fraction of **positive** examples **correctly** classified as positive
- FP rate =  $FP / (FP + TN)$  = 1 – specificity  
fraction of **negative** examples **incorrectly** classified as positive
- Performance of each classifier represented as a point on the ROC curve
- Changing the threshold of the algorithm changes the location of the point



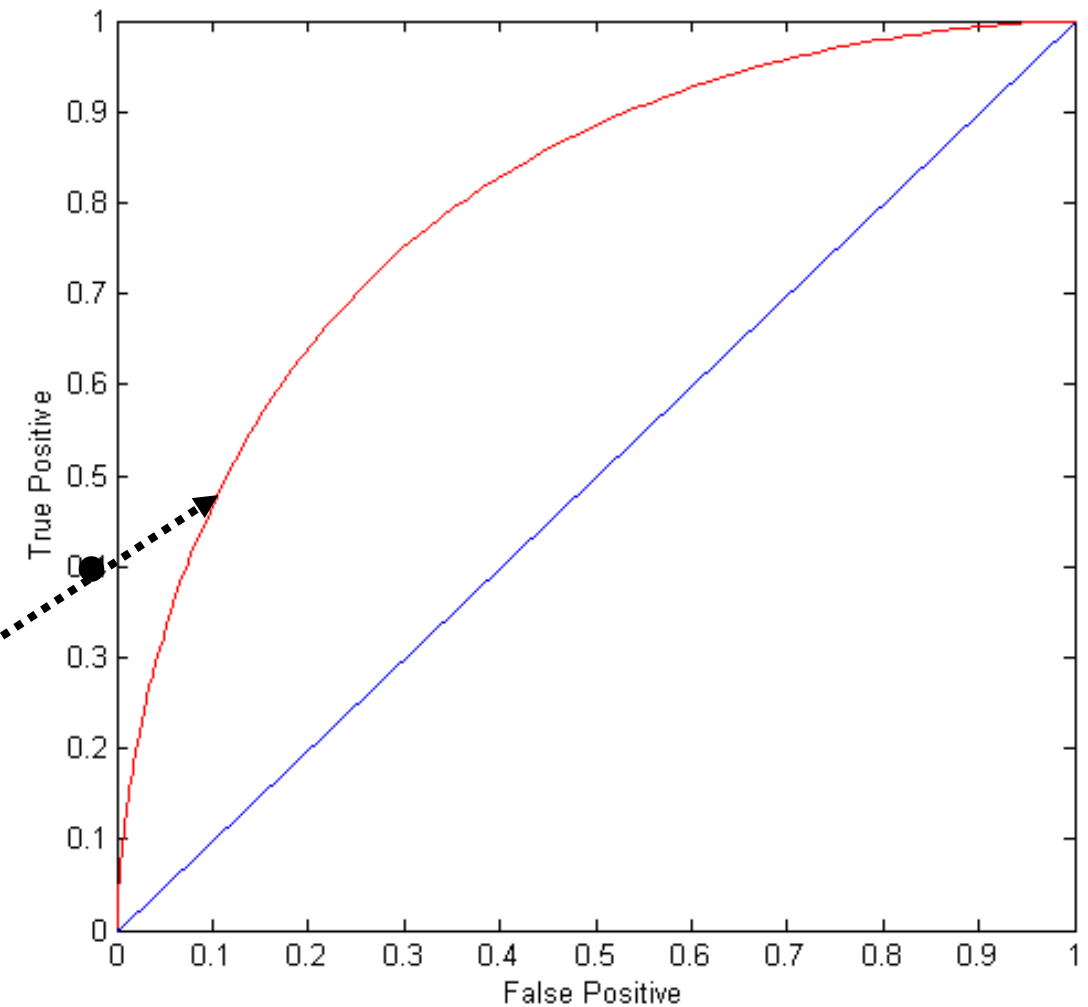
## ROC Curve (1)

- One-dimensional data set containing 2 classes (positive and negative)
- Points located at  $x > t$  classified as positive



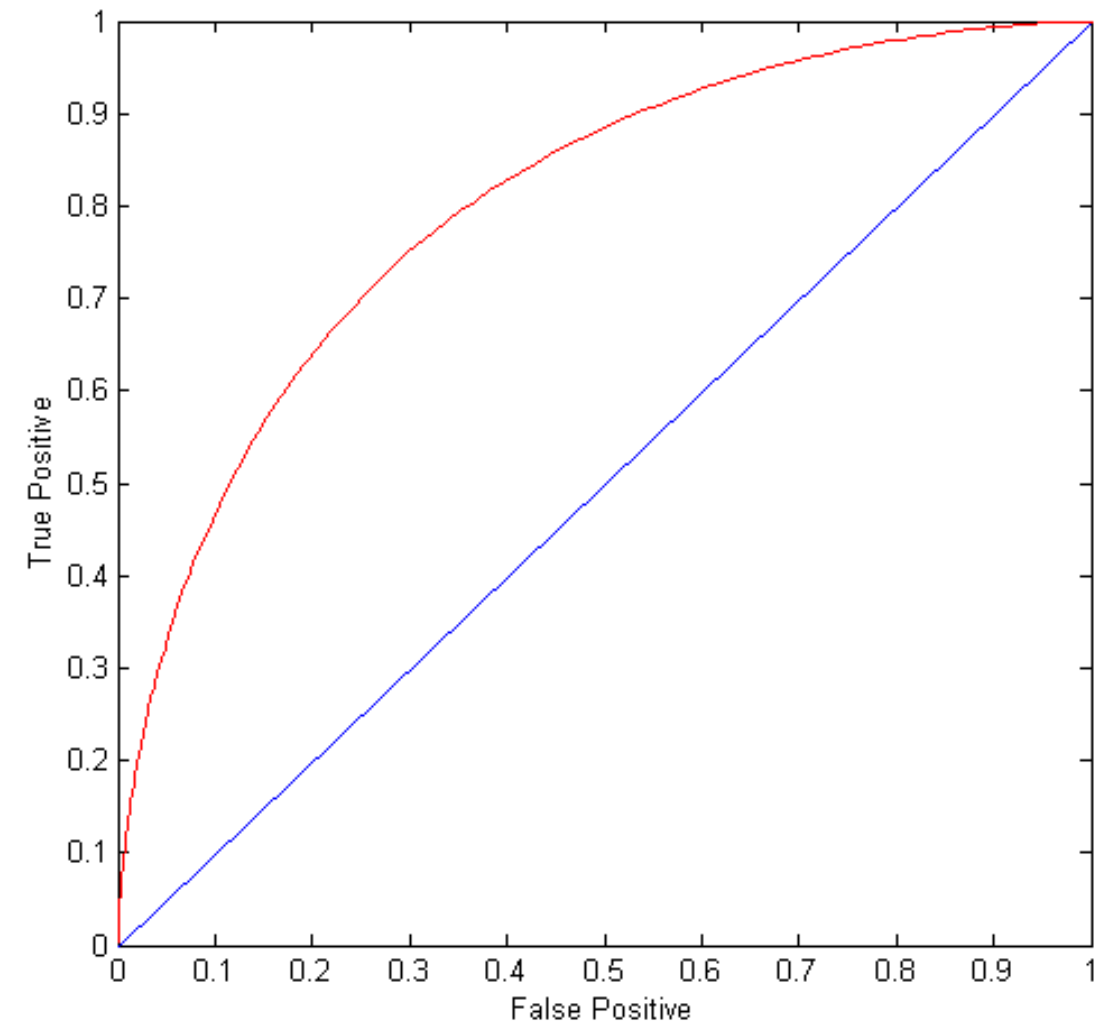
At threshold  $t$ :

TP rate=0.5, FN rate=0.5,  
FP rate=0.12, TN rate=0.88



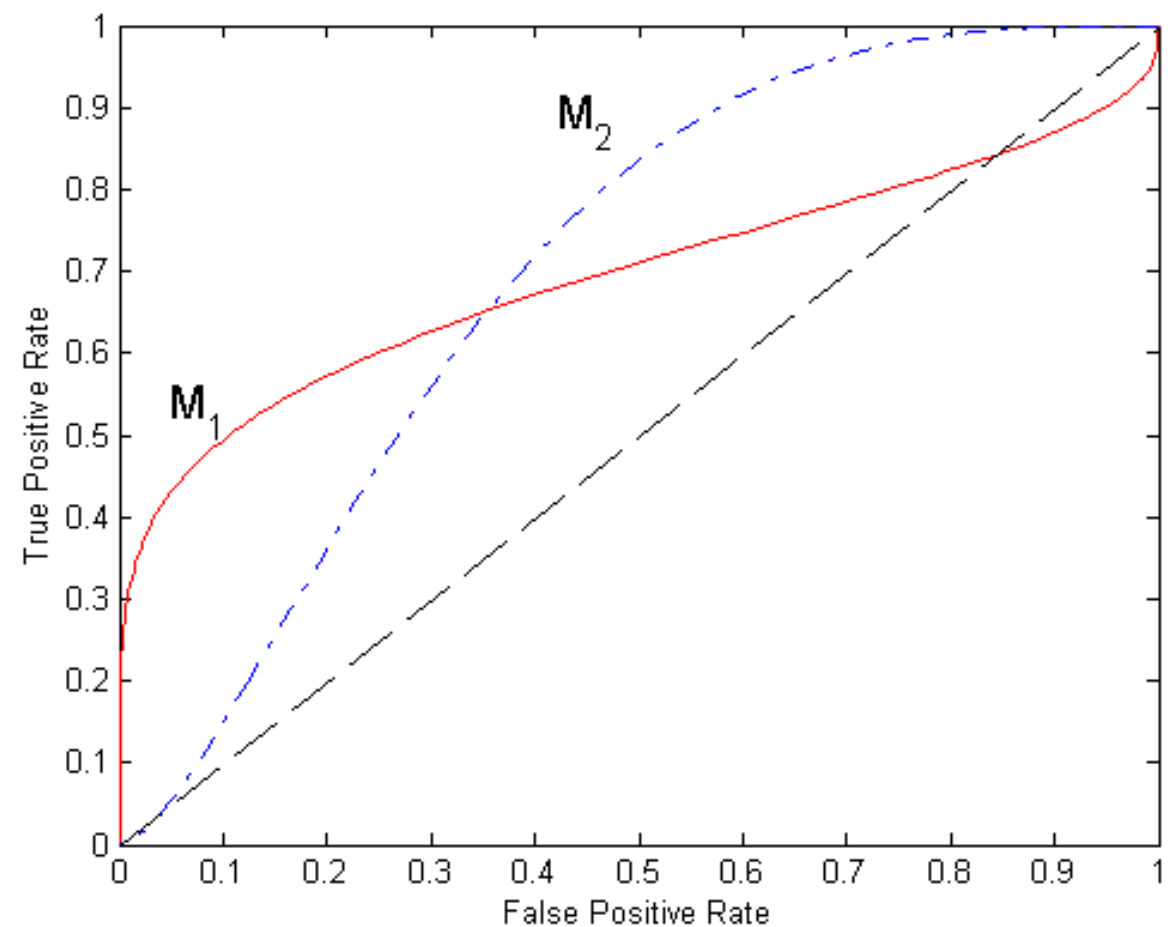
## ROC Curve (2)

- (TP,FP):
  - (0,0): declare everything to be negative class
  - (1,1): declare everything to be positive class
  - (1,0): ideal
- Diagonal line:
  - Random guessing
- Below diagonal line:
  - Prediction is opposite of the true class



# Using ROC for Model Comparison

- No model consistently outperforms the other
  - $M_1$  is better for small FPR
  - $M_2$  is better for large FPR
- Area Under the ROC Curve (AUC)
  - Ideal: area = 1
  - Random guess: area = 0.5



## How to Construct an ROC curve (1)

Instance	$P(+ A)$	True Class
1	1	+
2	0.93	+
3	0.87	-
4	0.85	+
5	0.85	-
6	0.85	-
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

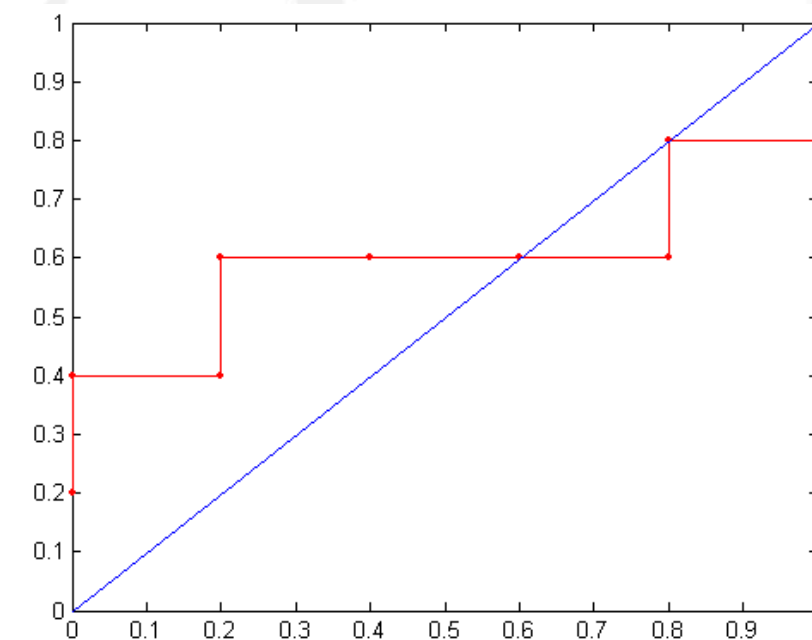
- Use classifier that produces posterior probability for each test instance  $P(+|A)$
- Sort the instances according to  $P(+|A)$  in decreasing order
- Apply threshold at each unique value of  $P(+|A)$
- Count the number of TP, FP, TN, FN at each threshold
- TP rate,  $\text{TPR} = \text{TP}/(\text{TP} + \text{FN})$
- FP rate,  $\text{FPR} = \text{FP}/(\text{FP} + \text{TN})$

## How to Construct an ROC curve (2)

Instance	$P(+ A)$	True Class
1	1	+
2	0.93	+
3	0.87	-
4	0.85	+
5	0.85	-
6	0.85	-
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

Class	+	-	+	-	-	-	+	-	+	+	
Threshold $\geq$	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC Curve:



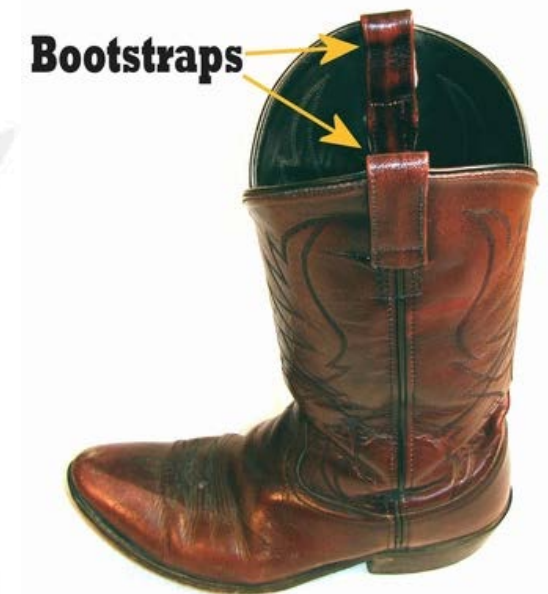
# Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?



# Methods of Estimation

- Holdout
  - Reserve, e.g., 2/3 for training and 1/3 for testing
  - Fine for huge data sets
- Random subsampling
  - Repeated holdout
- Cross-validation
  - Partition data into  $k$  disjoint subsets
  - $k$ -fold: train on  $k-1$  partitions, test on the remaining one
  - Leave-one-out:  $k=n$  with  $n$  the number of samples
  - Better than holdout for small data sets
- Bootstrap
  - Sampling with replacement



“pull oneself over a fence  
by one's bootstraps”

# Cross-Validation

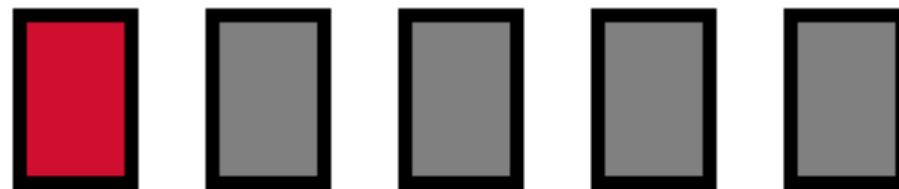
- Advantage over holdout: all data used for training and testing
- Predictions on test folds can be concatenated and can then be treated as if they are predictions on an independent test set, e.g., for making an ROC curve
- Ideal for comparing classifiers on relatively small data sets
- Make sure to use the **same splits** for all classifiers
- Can make use of **sign test** to check significance of difference in performance between classifiers (more later)

## Cross-validation Example

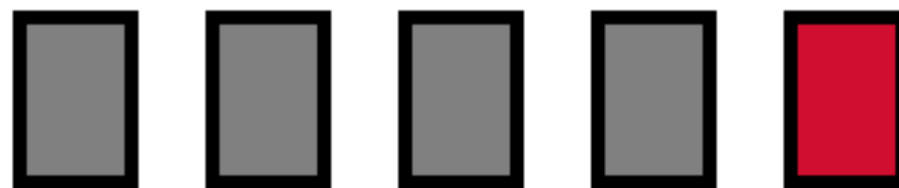
- Break up data into groups of the same size



- Hold aside one group for testing and use the rest to build model



- Repeat



## Stratified Ten-fold Cross-validation

- Standard method for evaluation: stratified ten-fold cross-validation
- Why ten? Works fine in practice and not too computationally intensive
- Stratification, i.e., keeping the same class distribution across folds, reduces the estimate's variance
- Even better: repeated stratified cross-validation
  - E.g. ten-fold cross-validation is repeated ten times and results are averaged (reduces the variance)

# The Bootstrap

- CV uses sampling without replacement
  - The same instance, once selected, cannot be selected again for a particular training/test set
- The bootstrap uses sampling with replacement to form the training set
  - Sample a dataset of  $n$  instances  $n$  times with replacement to form a new dataset of  $n$  instances
  - Use this data as the training set
  - Use the instances from the original dataset that don't occur in the new training set for testing
- A particular instance has a probability of  $1-1/n$  of not being picked
  - Thus its probability of ending up in the test data is:  $\left(1 - \frac{1}{n}\right)^n \approx e^{-1} = 0.368$
  - This means the training data will contain approximately 63.2% of the instances

## Estimating Error with the Bootstrap

- The error estimate on the test data will be very pessimistic
  - Trained on just ~63% of the instances
- Therefore, combine it with the training error:

$$err = 0.632 \cdot e_{\text{test}} + 0.368 \cdot e_{\text{training}}$$

- The training error gets less weight than the error on the test data
- Repeat process several times with different replacement samples; average the results
- Referred to as the 0.632 bootstrap estimator

## General Issues

- Unbalanced data
- Tuning “hyperparameters”
- Learning curves

Nice overview with recommendations:

Raschka, S. (2018). Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning. *arXiv:1811.12808v2*.

# Unbalanced Data

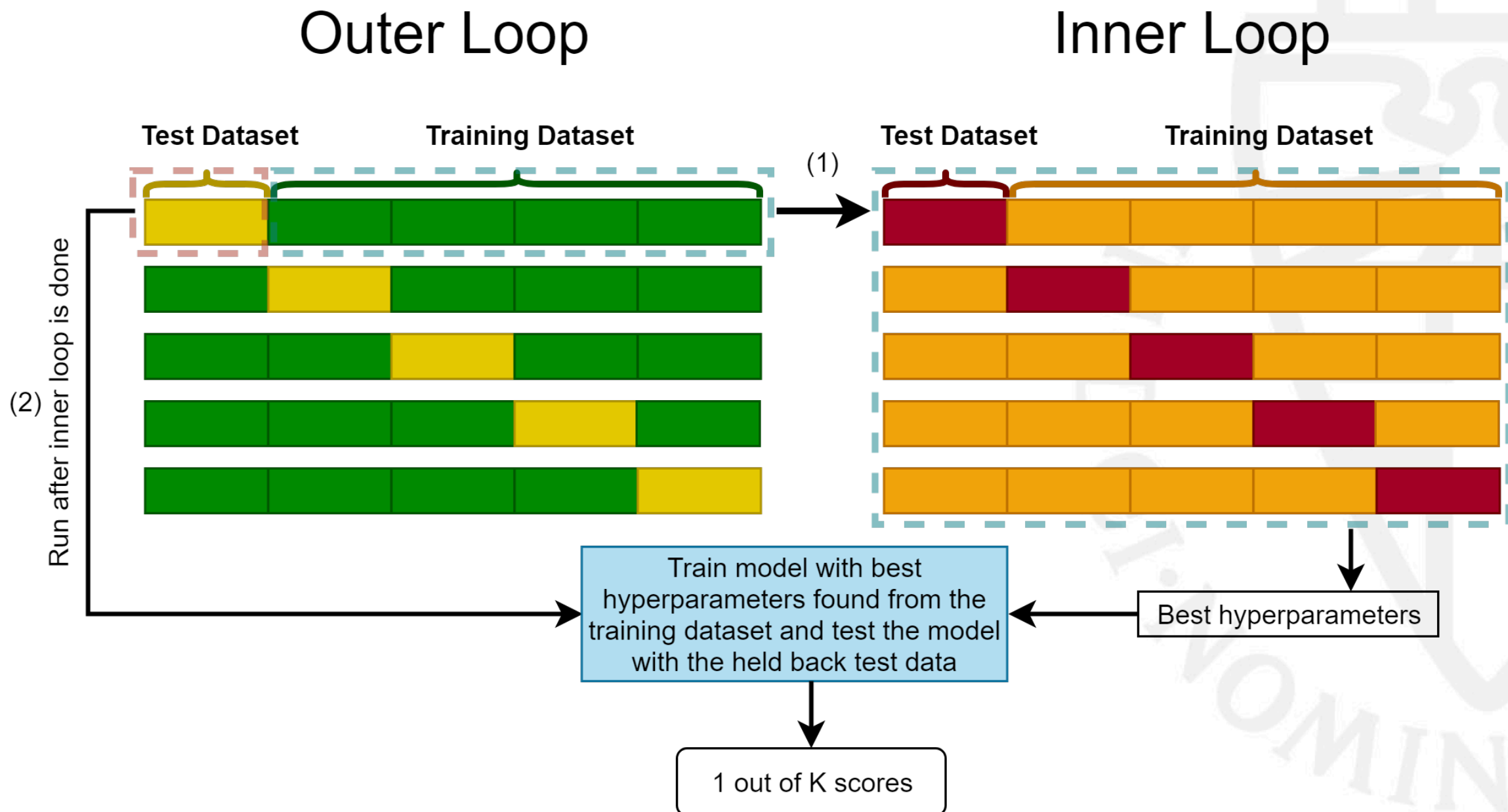
- Sometimes, classes have very unequal frequency
  - medical diagnosis: 90% healthy, 10% disease
  - eCommerce: 99% don't buy, 1% buy
  - Security: >99.99% of Americans are not terrorists
- Majority class classifier can be very close to 100% correct, but is still completely useless...
- Measuring performance by ROC at least helps a bit
- Alternative is to construct **balanced** train and test sets, and train model on a balanced set
  - consider all minority class instances
  - add equal number of randomly selected samples from the majority class



# Tuning Hyperparameters

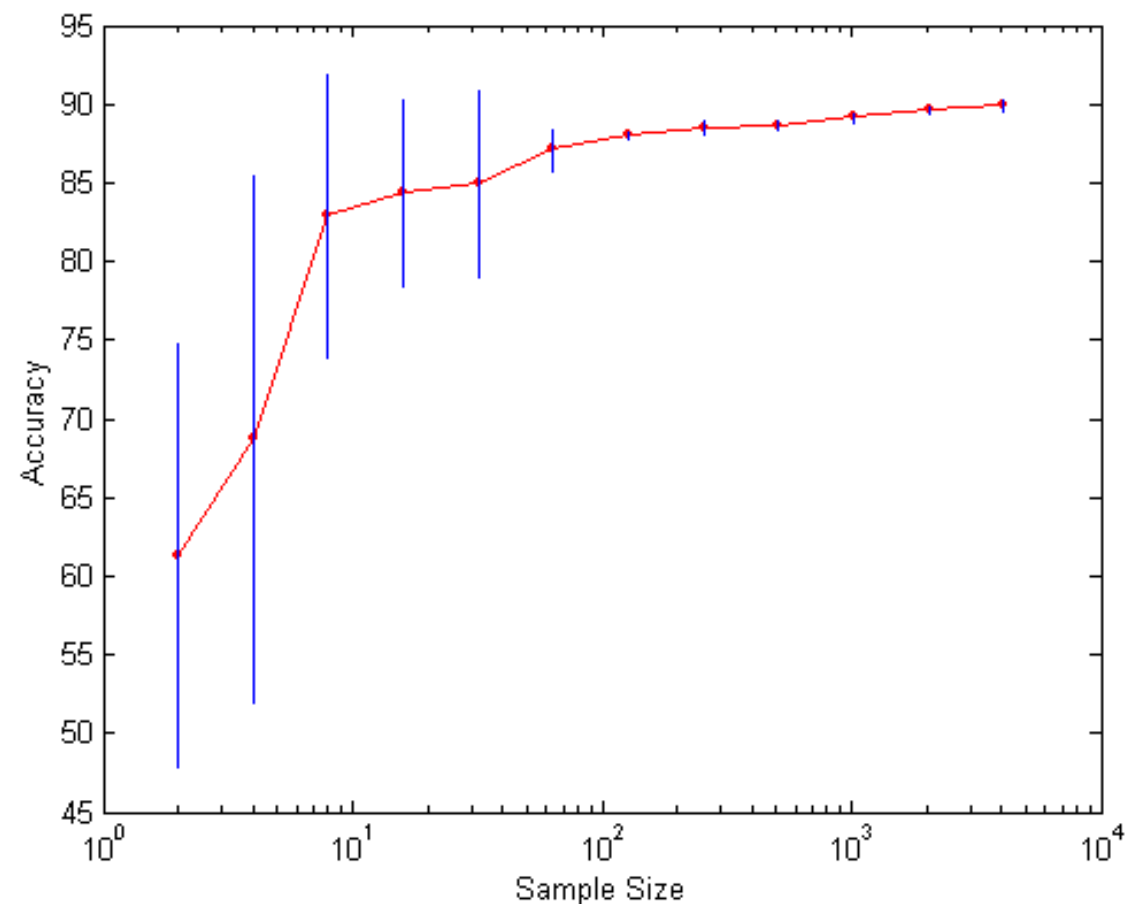
- Many classification methods contain “**hyperparameters**”, e.g., the maximum depth of the tree, the number of hidden units in a neural network, the regularization parameter in support vector machines, the number of nearest neighbors in  $k$  nearest neighbors
- Test data should not be used in any way to tune the hyperparameters!
- Proper procedure uses **nested cross-validation** with  
“inner” cross-validation to optimize the hyperparameters and  
“outer” cross-validation to estimate the generalization of the whole procedure (including the optimization of the hyperparameters)

# Nested cross-validation



# Learning Curves

- **Learning curve** shows how accuracy changes with varying sample size
- Can be constructed by **subsampling** smaller data sets from the full data set
- Useful to check whether collecting more data may still further improve performance



# Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- **Methods for Model Comparison**
  - How to compare the relative performance among competing models?

## Two Classifiers, Single Test Set

- Given a test set with the true class and the predictions of the two models:

Instance	$M_1$	$M_2$	True Class
1	+	+	+
2	+	+	+
3	+	-	-
4	-	-	-
5	-	-	-
6	+	-	+
7	-	+	-
8	-	-	+
9	+	+	-
...	...	....	...

- Translate to table with correct/wrong:

		$M_2$	
		correct	wrong
$M_1$	correct	50	6
	wrong	14	30
		64	36

**NB:** this part deviates from the book, which considers two classifiers on different test sets, which is silly and hardly ever occurs in practice

## Example 1

		$M_2$		
$M_1$		correct	wrong	
	correct	50	9	59
	wrong	11	30	41
		61	39	

- Is  $M_2$  significantly better than  $M_1$  at significance level 0.05?

## Example 2

		$M_2$		
		correct	wrong	
$M_1$	correct	5000	900	5900
	wrong	1100	3000	4100
		6100	3900	

- Is  $M_2$  significantly better than  $M_1$  at significance level 0.05?

## Example 3

		$M_2$		
$M_1$		correct	wrong	
	correct	50	0	50
	wrong	10	40	50
		60	40	

- Is  $M_2$  significantly better than  $M_1$  at significance level 0.05?



## Example 4

		$M_2$		
		correct	wrong	
$M_1$	correct	80	0	80
	wrong	10	10	20
		90	10	

- Is  $M_2$  significantly better than  $M_1$  at significance level 0.05?

## Example 5

		$M_2$		
$M_1$		correct	wrong	
	correct	5	0	5
	wrong	2	3	5
		7	3	

- Is  $M_2$  significantly better than  $M_1$  at significance level 0.05?

## Example 6

		$M_2$		
$M_1$		correct	wrong	
	correct	50	6	56
	wrong	14	30	44
		64	36	

- Is  $M_2$  significantly better than  $M_1$  at significance level 0.05?

## Sign Test (1)

- General test for **difference in the median** between **paired** samples
- Non-parametric: makes no assumptions about distribution
- $x_1 = 1$  if  $M_1$  is correct,  $x_1 = 0$  if  $M_1$  is wrong; similar for  $x_2$
- Is the median of  $x_2$  larger than the median of  $x_1$ ?
- **Null hypothesis**: for a random pair of samples,  $x_1$  and  $x_2$  are equally likely to be larger than the other

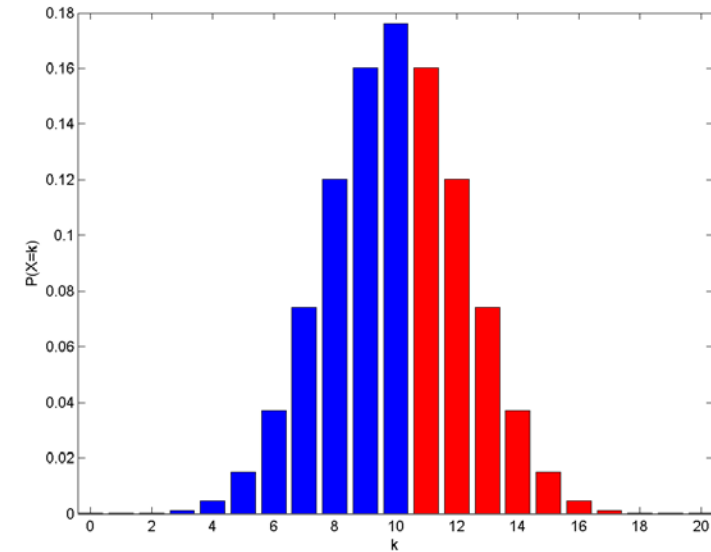
## Sign Test (2)

- Call  $N_{1<2}$  the number of samples for which  $M_2$  is correct and  $M_1$  is incorrect
- Samples for which  $x_1 = x_2$  can be omitted: only  $N_{1<2}$  and  $N_{1>2}$  matter!
- Boils down to a **binomial** test:
  - Null hypothesis: fair “coin”,  $p = 0.5$
  - Test statistic:  $W = N_{1<2}$
  - Under the null hypothesis, the test statistic follows a binomial distribution with number of draws  $N = N_{1<2} + N_{1>2}$
- One-sided alternative:  $M_2 > M_1$ , i.e.,  $p > 0.5$
- Two-sided alternative:  $M_2 \neq M_1$ , i.e.,  $p \neq 0.5$
- A.k.a. an exact version of **McNemar**’s test (the original one uses a chi-squared test, which is a fine approximation for  $N$  large)

		$M_2$	
		correct	wrong
$M_1$	correct	??	$N_{1>2}$
	wrong	$N_{1<2}$	??

## Example 1

		$M_2$		
		correct	wrong	
$M_1$	correct	50	9	59
	wrong	11	30	41
		61	39	



$\text{Binocdf}(10, 20, 0.5)$

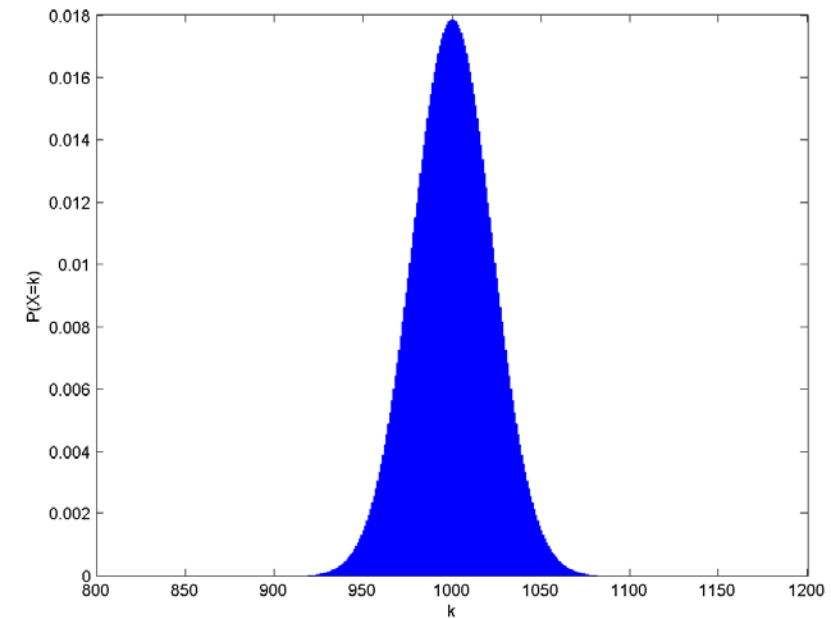
$P(W \geq 11)$

- $N = 20$  draws
- P-value =  $P(W \geq 11) = 1 - \text{Binocdf}(10, 20, 0.5) = 0.4199 > 0.05$
- Clearly **not significant**

null hypothesis:  $p = 0.5$

## Example 2

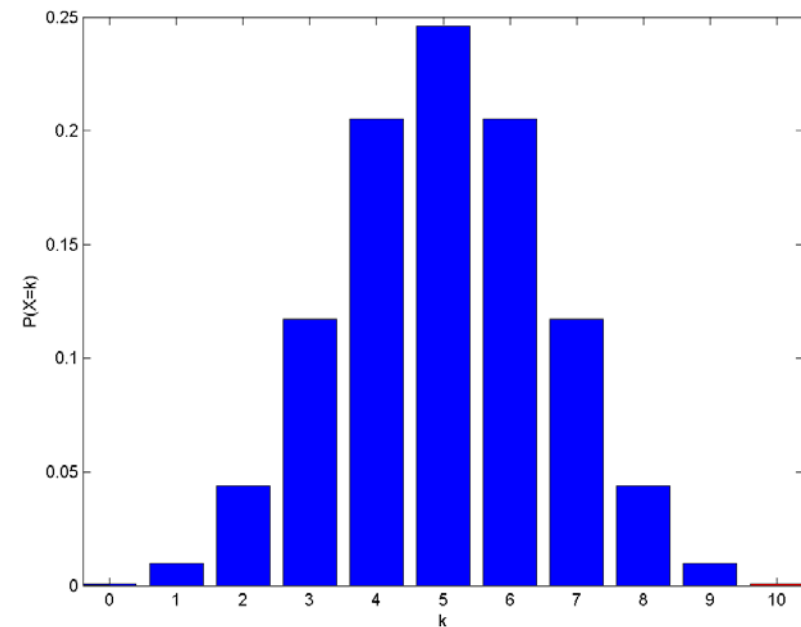
		$M_2$		
		correct	wrong	
$M_1$	correct	5000	900	5900
	wrong	1100	3000	4100
		6100	3900	



- $N = 2000$  draws
- $P\text{-value} = P(W \geq 1100) = 1 - \text{Binocdf}(1099, 2000, 0.5) = 4.23 \times 10^{-6} < 0.05$
- Clearly **significant**

## Example 3

		$M_2$		
		correct	wrong	
$M_1$	correct	50	0	50
	wrong	10	40	50
		60	40	

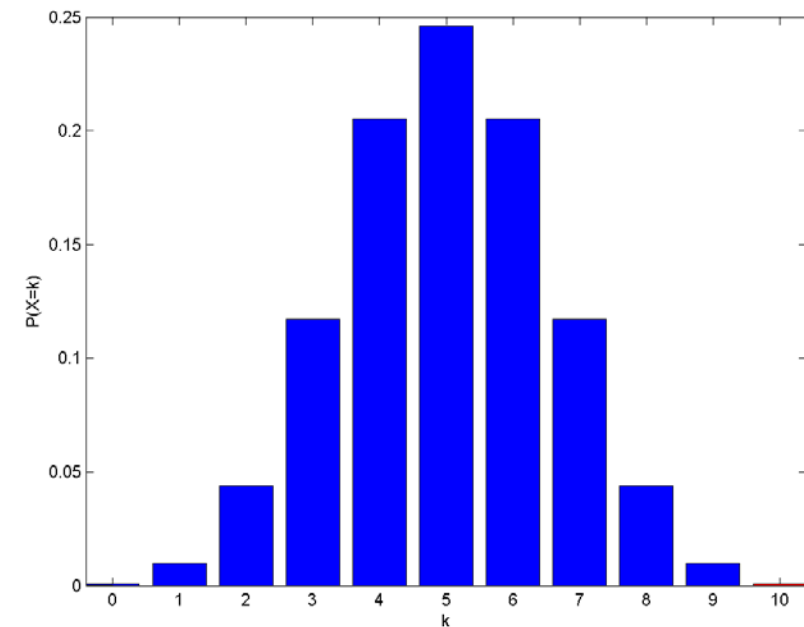


- $N = 10$  draws
- P-value =  $P(W \geq 10) = 1 - \text{Binocdf}(9, 10, 0.5) = 9.77 \times 10^{-4} (= 2^{-10}) < 0.05$
- Clearly **significant**



## Example 4

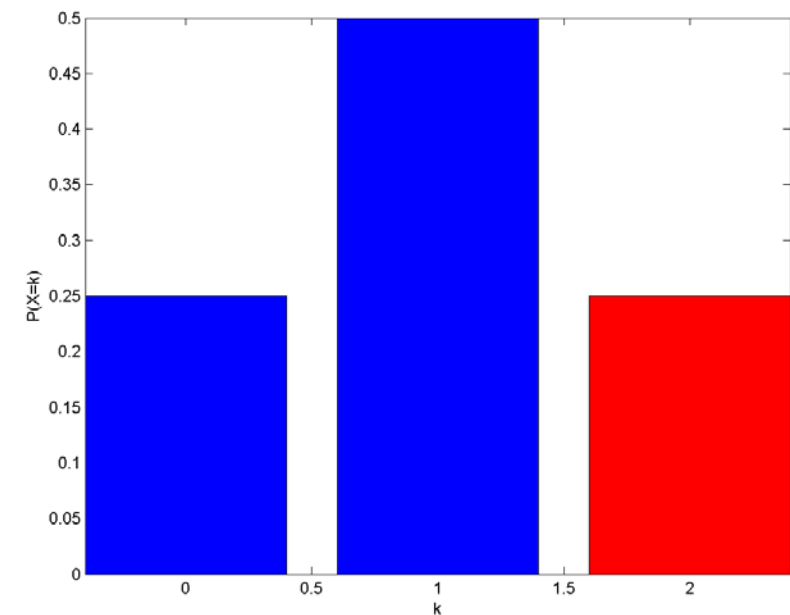
		$M_2$		
		correct	wrong	
$M_1$	correct	80	0	80
	wrong	10	10	20
		90	10	



- $N = 10$  draws: same as before
- P-value =  $P(W \geq 10) = 1 - \text{Binocdf}(9, 10, 0.5) = 9.77 \times 10^{-4} (= 2^{-10}) < 0.05$
- Clearly **significant**

## Example 5

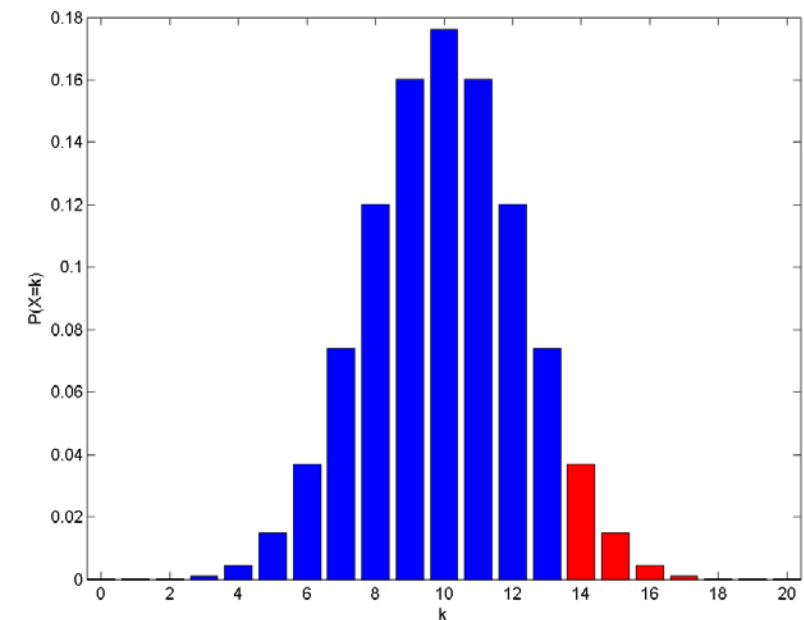
		$M_2$		
		correct	wrong	
$M_1$	correct	5	0	5
	wrong	2	3	5
		7	3	



- $N = 2$  draws
- P-value =  $P(W \geq 2) = 1 - \text{Binocdf}(1, 2, 0.5) = 0.25 (= 2^{-2}) > 0.05$
- Clearly **not significant**

## Example 6

		$M_2$		
		correct	wrong	
$M_1$	correct	50	6	56
	wrong	14	30	44
		64	36	



- $N = 20$  draws
- $P\text{-value} = P(W \geq 14) = 1 - \text{Binocdf}(13, 20, 0.5) = 0.0577 > 0.05$
- Just **not significant**

## Two Classifiers, Multiple Test Sets

- Standard practice: compare different classifiers on a whole range of problems, e.g., from the UCI repository
- Simplest idea: use the sign/McNemar's test, but now on **complete test sets** instead of test samples
- Now  $N_{1<2}$  is the number of test sets on which  $M_2$  does better than  $M_1$
- The **Wilcoxon signed-rank test** also takes into account how much better a classifier does
- Demšar, JMLR 2006, is the key reference, also for comparing multiple classifiers on multiple test sets

Journal of Machine Learning Research 7 (2006) 1–30

Submitted 8/04; Revised 4/05; Published 1/06

### Statistical Comparisons of Classifiers over Multiple Data Sets

Janez Demšar

Faculty of Computer and Information Science  
Tržaška 25  
Ljubljana, Slovenia

JANEZ.DEMSAR@FRI.UNI-LJ.SI