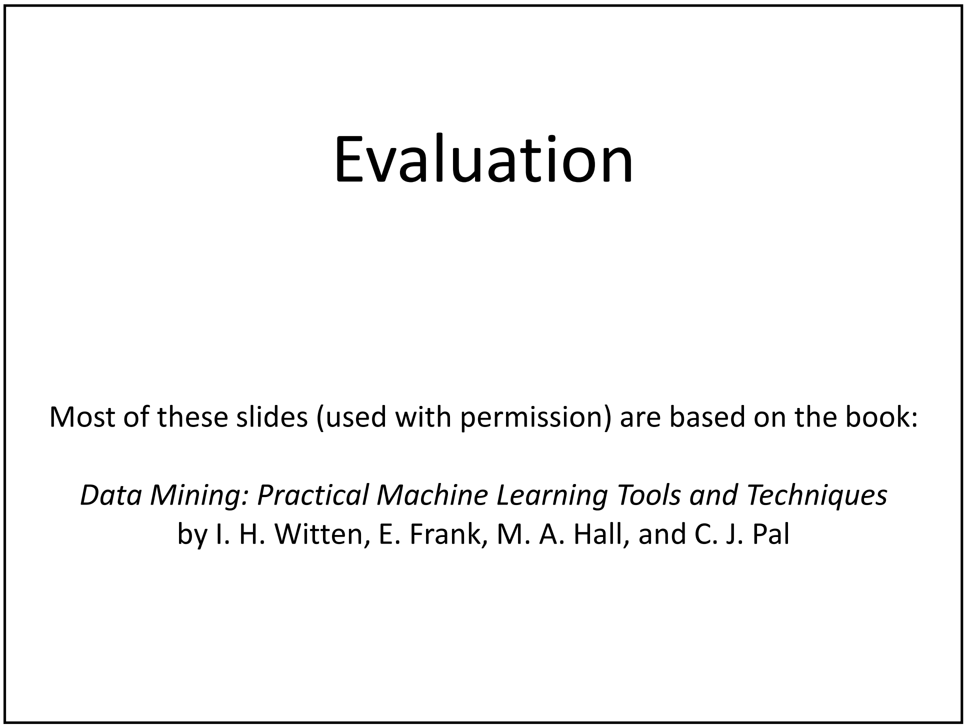
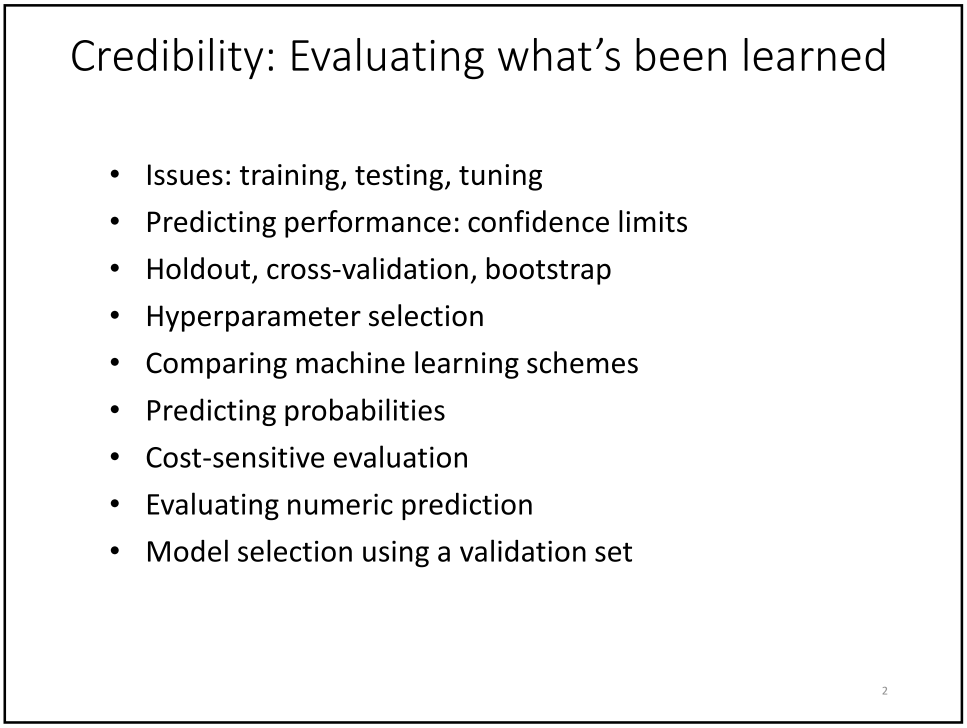


Evaluation

Most of these slides (used with permission) are based on the book:

Data Mining: Practical Machine Learning Tools and Techniques
by I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal

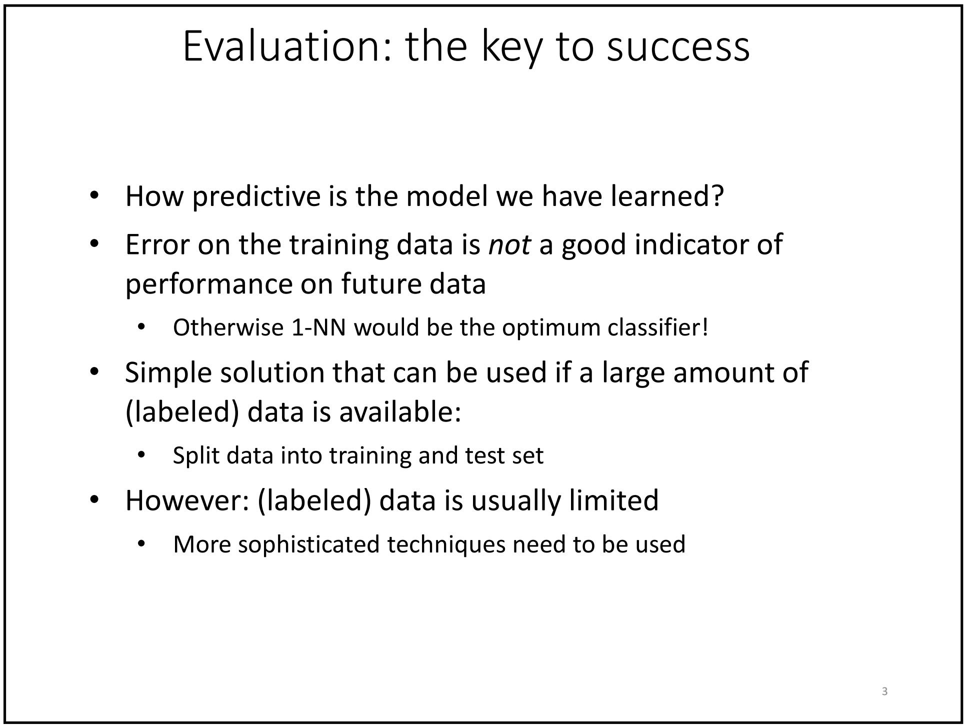
1



Credibility: Evaluating what’s been learned

- Issues: training, testing, tuning
- Predicting performance: confidence limits
- Holdout, cross-validation, bootstrap
- Hyperparameter selection
- Comparing machine learning schemes
- Predicting probabilities
- Cost-sensitive evaluation
- Evaluating numeric prediction
- Model selection using a validation set

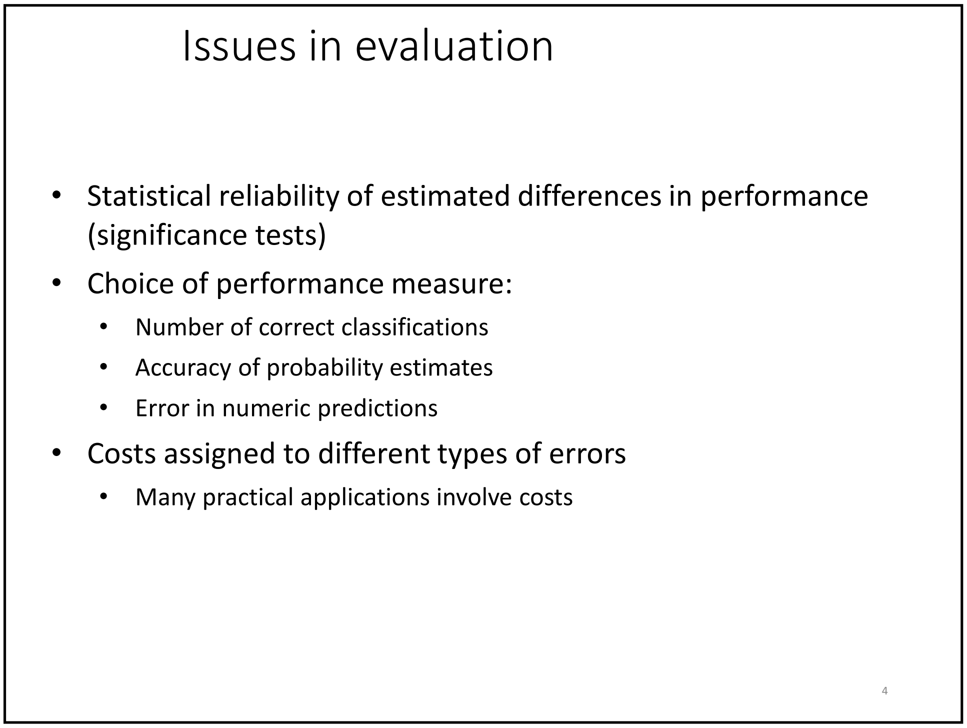
2



Evaluation: the key to success

- How predictive is the model we have learned?
- Error on the training data is *not* a good indicator of performance on future data
  - Otherwise 1-NN would be the optimum classifier!
- Simple solution that can be used if a large amount of (labeled) data is available:
  - Split data into training and test set
- However: (labeled) data is usually limited
  - More sophisticated techniques need to be used

3



Issues in evaluation

- Statistical reliability of estimated differences in performance (significance tests)
- Choice of performance measure:
  - Number of correct classifications
  - Accuracy of probability estimates
  - Error in numeric predictions
- Costs assigned to different types of errors
  - Many practical applications involve costs

4

Training and testing I

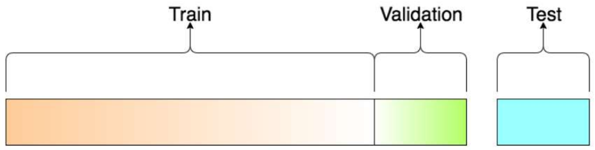
- Natural performance measure for classification problems: *error rate*
  - *Success*: instance’s class is predicted correctly
  - *Error*: instance’s class is predicted incorrectly
  - Error rate: proportion of errors made over the whole set of instances
- *Resubstitution error:* error rate obtained by evaluating model on training data
- Resubstitution error is (hopelessly) optimistic!

Training and testing II

- *Test set*: independent instances that have played no part in formation of classifier
  - Assumption: both training data and test data are representative samples of the underlying problem
- Test and training data may differ in nature
  - Example: classifiers built using customer data from two different towns *A* and *B*
  - To estimate performance of classifier from town *A* in completely new town, test it on data from *B*

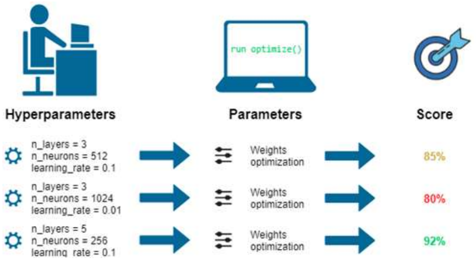
Note on parameter tuning

- It is important that the test data is not used *in any way* to create the classifier
- Some learning schemes operate in two stages:
  - Stage 1: build the basic structure
  - Stage 2: optimize parameter settings
- The test data cannot be used for parameter tuning!
- Proper procedure uses *three* sets: *training data*, *validation data*, and *test data*
  - Validation data is used to optimize parameters



Examples of hyper parameters

- Neural networks



n_layers = 3
n_neurons = 512
learning_rate = 0.1

Weights optimization

85%

n_layers = 3
n_neurons = 1024
learning_rate = 0.01

Weights optimization

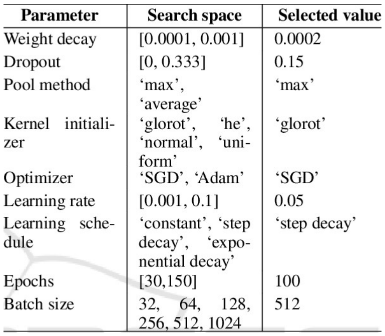
80%

n_layers = 5
n_neurons = 256
learning_rate = 0.1

Weights optimization

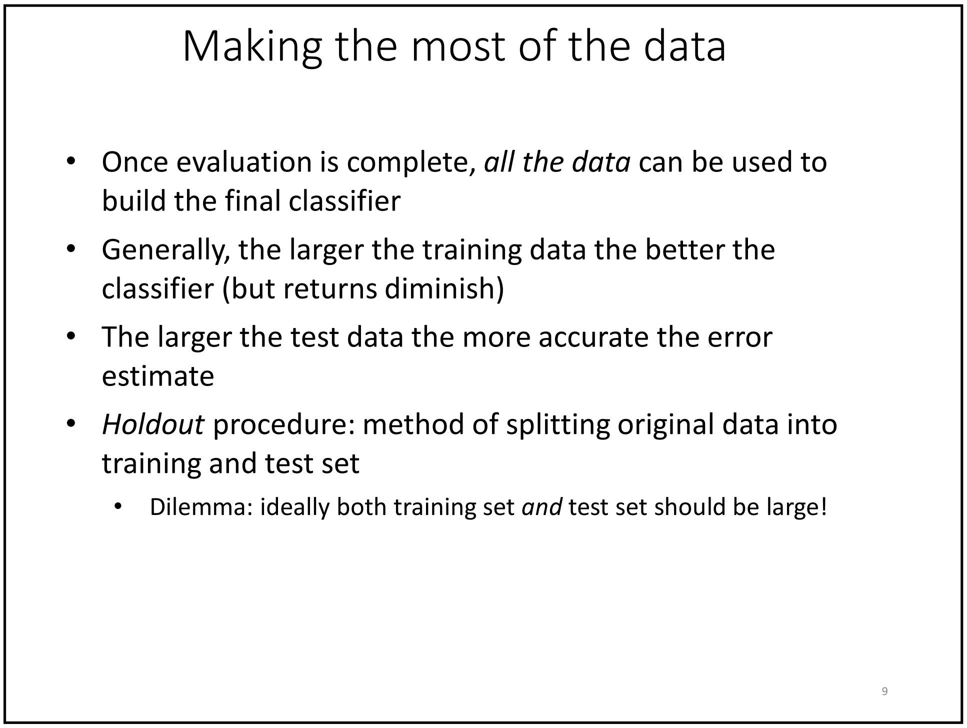
92%

- Deep neural networks



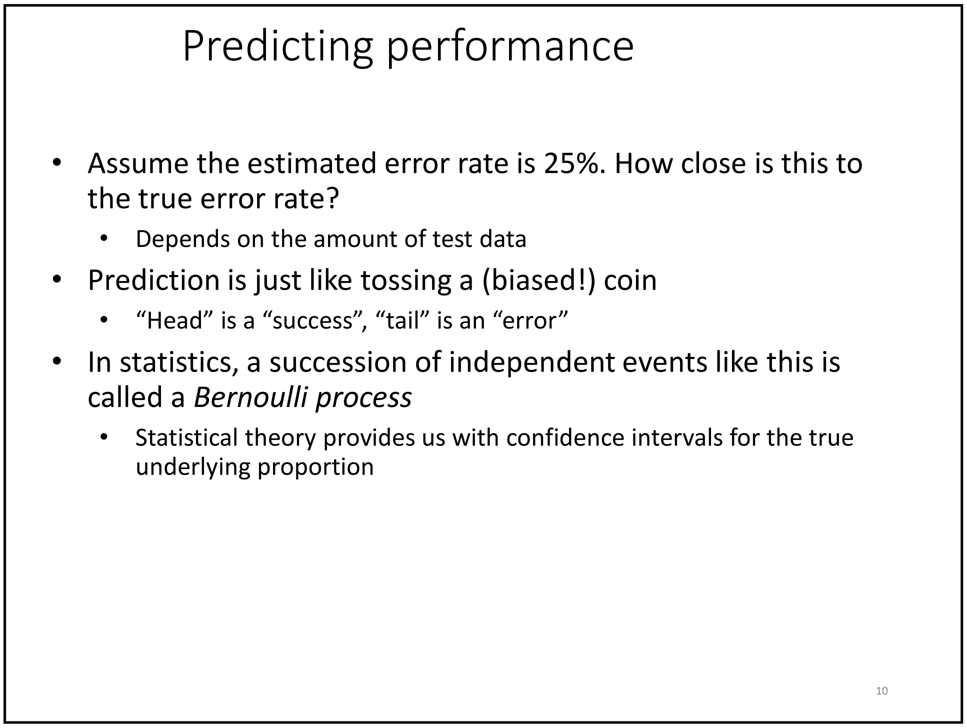
Parameter	Search space	Selected value
Weight decay	[0.0001, 0.001]	0.0002
Dropout	[0, 0.333]	0.15
Pool method	'max', 'average'	'max'
Kernel initializer	'glorot', 'he', 'normal', 'uniform'	'glorot'
Optimizer	'SGD', 'Adam'	'SGD'
Learning rate	[0.001, 0.1]	0.05
Learning schedule	'constant', 'step decay', 'exponential decay'	'step decay'
Epochs	[30,150]	100
Batch size	32, 64, 128, 256, 512, 1024	512

- Decision tree hyper parameters
  - Max depth of the tree
  - Minimum number of samples in leaf node
  - Use reduced error pruning



Making the most of the data

- Once evaluation is complete, *all the data* can be used to build the final classifier
- Generally, the larger the training data the better the classifier (but returns diminish)
- The larger the test data the more accurate the error estimate
- *Holdout* procedure: method of splitting original data into training and test set
  - Dilemma: ideally both training set *and* test set should be large!



Predicting performance

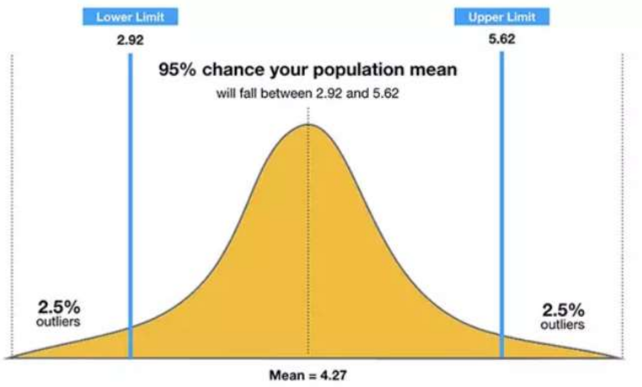
- Assume the estimated error rate is 25%. How close is this to the true error rate?
  - Depends on the amount of test data
- Prediction is just like tossing a (biased!) coin
  - “Head” is a “success”, “tail” is an “error”
- In statistics, a succession of independent events like this is called a *Bernoulli process*
  - Statistical theory provides us with confidence intervals for the true underlying proportion

Confidence intervals

- *p* refers to a success rate
- We can say: *p* lies within a certain specified interval with a certain specified confidence
  - Example: *S*=750 successes in *N*=1000 trials
  - Estimated success rate: 75%
  - How close is this to true success rate *p*?
  - Answer: with 80% confidence *p* is located in [73.2,76.7]
- Another example: *S*=75 and *N*=100
  - Estimated success rate: 75%
  - With 80% confidence *p* in [69.1,80.1]

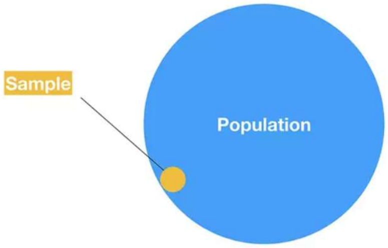
Confidence intervals

- A confidence interval (CI) is a range of values that's likely to include a population value with a certain degree of confidence.
- It is often expressed a % whereby a population means lies between an upper and lower interval.



Why confidence interval is used?

- It is more or less impossible to study every single person in a population so researchers select a sample or sub-group of the population.
- A confidence interval is simply a way to measure how well your sample represents the population you are studying.
- See the example: https://www.simplypsychology.org/confidence-interval.html



Holdout estimation

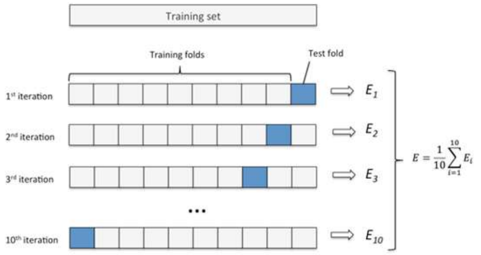
- What should we do if we only have a single dataset?
- The *holdout* method reserves a certain amount for testing and uses the remainder for training, after shuffling
  - Usually: one third for testing, the rest for training
- Problem: the samples might not be representative
  - Example: class might be missing in the test data
- Advanced version uses *stratification*
  - Ensures that each class is represented with approximately equal proportions in both subsets

Repeated holdout method

- Holdout estimate can be made more reliable by repeating the process with different subsamples
  - In each iteration, a certain proportion is randomly selected for training (possibly with stratificiation)
  - The error rates on the different iterations are averaged to yield an overall error rate
- This is called the *repeated holdout* method
- Still not optimum: the different test sets overlap
  - Can we prevent overlapping?

Cross-validation

- *K-fold cross-validation* avoids overlapping test sets
  - First step: split data into *k* subsets of equal size
  - Second step: use each subset in turn for testing, the remainder for training
  - This means the learning algorithm is applied to *k* different training sets
- Often the subsets are stratified before the cross-validation is performed to yield stratified *k*-fold cross-validation
- The error estimates are averaged to yield an overall error estimate; also, standard deviation is often computed
- Alternatively, predictions and actual target values from the *k* folds are pooled to compute one estimate



More on cross-validation

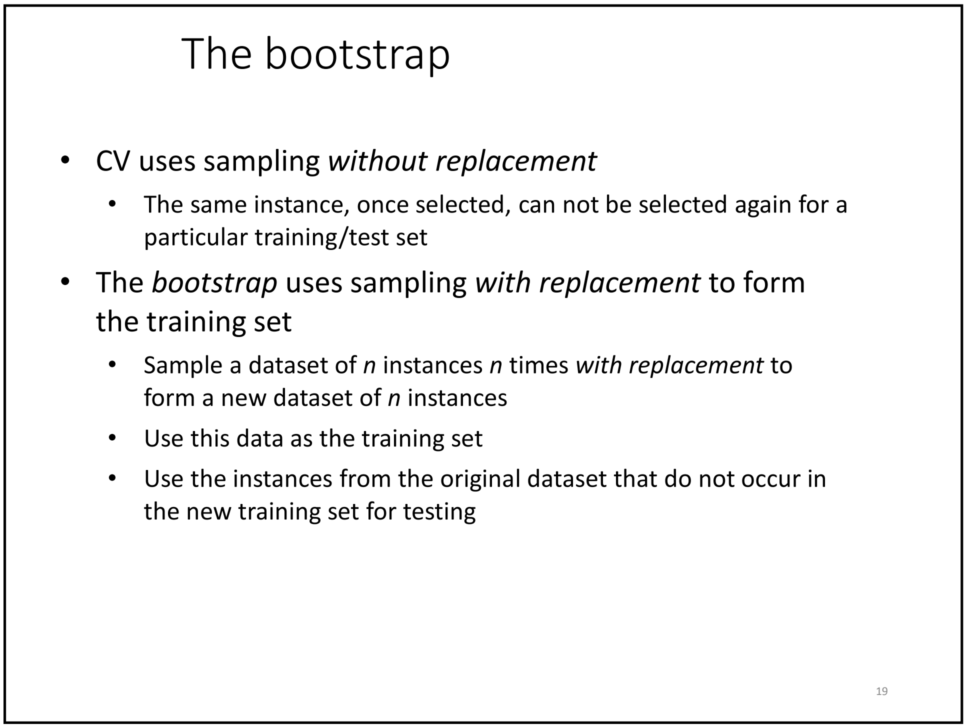
- Standard method for evaluation: stratified ten-fold cross-validation
- Why ten?
  - Extensive experiments have shown that this is the best choice to get an accurate estimate
  - There is also some theoretical evidence for this
- Stratification reduces the estimate's variance
- Even better: repeated stratified cross-validation
  - E.g., ten-fold cross-validation is repeated ten times and results are averaged (reduces the variance)

17

Leave-one-out cross-validation

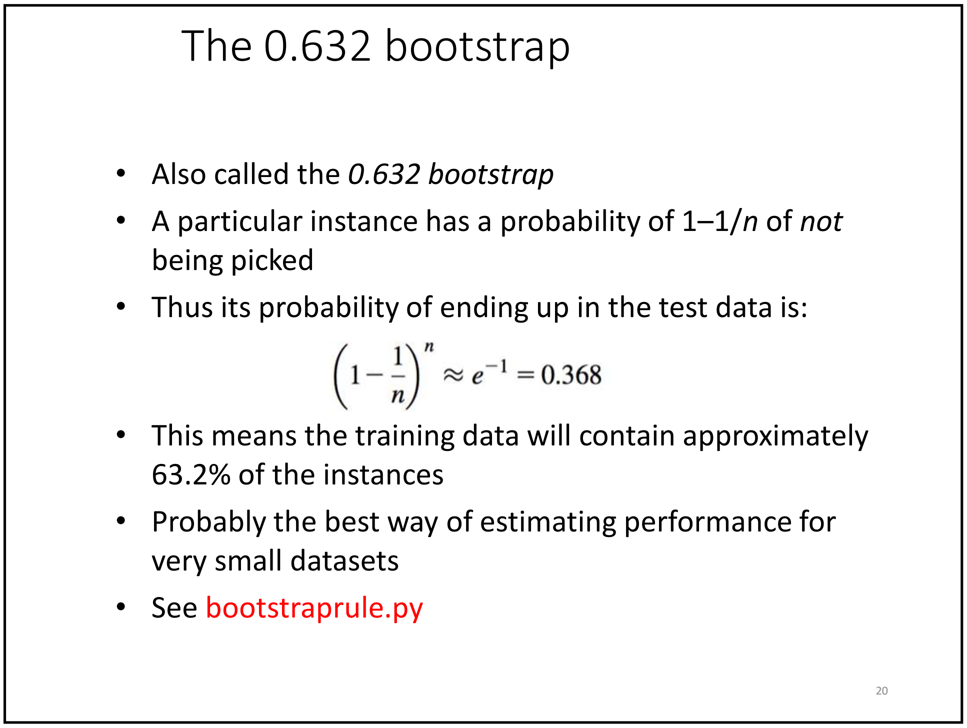
- Leave-one-out:
  a particular form of *k*-fold cross-validation:
  - Set number of folds to number of training instances
  - I.e., for *n* training instances, build classifier *n* times
- Makes best use of the data
- Involves no random subsampling
- Very computationally expensive
- Disadvantage of Leave-one-out CV: stratification is not possible
  - It *guarantees* a non-stratified sample because there is only one instance in the test set!

18



The bootstrap

- CV uses sampling *without replacement*
  - The same instance, once selected, can not 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 do not occur in the new training set for testing



The 0.632 bootstrap

- Also called the *0.632 bootstrap*
- 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
- Probably the best way of estimating performance for very small datasets
- See bootstraprule.py

Hyperparameter selection

- *Hyperparameter:* parameter that can be tuned to optimize the performance of a learning algorithm
  - Different from basic parameter that is part of a model, such as a coefficient in a linear regression model
  - Example hyperparameter: *k* in the *k*-nearest neighbour classifier
- We are not allowed to peek at the final test data to choose the value of this parameter
  - Adjusting the hyperparameter to the test data will lead to optimistic performance estimates on this test data!
  - Parameter tuning needs to be viewed as part of the learning algorithm and must be done using the training data only

Hyperparameters and cross-validation

- Note that *k*-fold cross-validation runs *k* different train-test evaluations
  - The above parameter tuning process using validation sets must be applied separately to each of the *k* training sets!
- This means that, when hyperparameter tuning is applied, *k* different hyperparameter values may be selected
  - This is OK: hyperparameter tuning is part of the learning process
  - Cross-validation evaluates the quality of the learning process, not the quality of a particular model

Comparing machine learning schemes

- Frequent question: which of two learning schemes performs better?
- Note: this is domain dependent!
- Obvious way: compare 10-fold cross-validation estimates
- Generally sufficient in applications (we do not loose if the chosen method is not truly better)
- However, what about machine learning research?
- Need to show convincingly that a particular method works better in a particular domain from which data is taken

Comparing learning schemes II

- Want to show that scheme A is better than scheme B in a particular domain
  - For a given amount of training data (i.e., data size)
  - On average, across all possible training sets from that domain
- Let's assume we have an infinite amount of data from the domain
- Then, we can simply
  - sample infinitely many dataset of a specified size
  - obtain a cross-validation estimate on each dataset for each scheme
  - check if the mean accuracy for scheme A is better than the mean accuracy for scheme B

Paired t-test

- In practice, we have limited data and a limited number of estimates for computing the mean
- *Student's t-test* tells us whether the means of two samples are significantly different
- In our case the samples are cross-validation estimates, one for each dataset we have sampled
- We can use a *paired* t-test because the individual samples are paired
  - The same cross-validation is applied twice, ensuring that all the training and test sets are exactly the same
  - A large t-score tells you that the groups are different
  - A small t-score tells you that the groups are similar
- Example: https://www.statisticshowto.datasciencecentral.com/probability-and-statistics/t-test/

William Gosset

Born: 1876 in Canterbury; Died: 1937 in Beaconsfield, England

Obtained a post as a chemist in the Guinness brewery in Dublin in 1899. Invented the t-test to handle small samples for quality control in brewing. Wrote under the name "Student".

Predicting probabilities

- Performance measure so far: success rate
- Also called *0-1 loss function*:

$$\sum_i \begin{cases} 0 \text{ if prediction is correct} \\ 1 \text{ if prediction is incorrect} \end{cases}$$

- Most classifiers produces class probabilities
- Depending on the application, we might want to check the accuracy of the probability estimates
- 0-1 loss is not the right thing to use in those cases

Quadratic loss function

- $p_1 \dots p_k$ are probability estimates for an instance
- c is the index of the instance’s actual class
- $a_1 \dots a_k = 0$, except for a_c which is 1
- *For single instance quadratic loss* is: $\sum_i (p_j - a_j)^2$
- For several instances we Want to minimize the following where *i* is the correct class.

$$1 - 2p_i + \sum_j p_j^2$$

27

Informational loss function

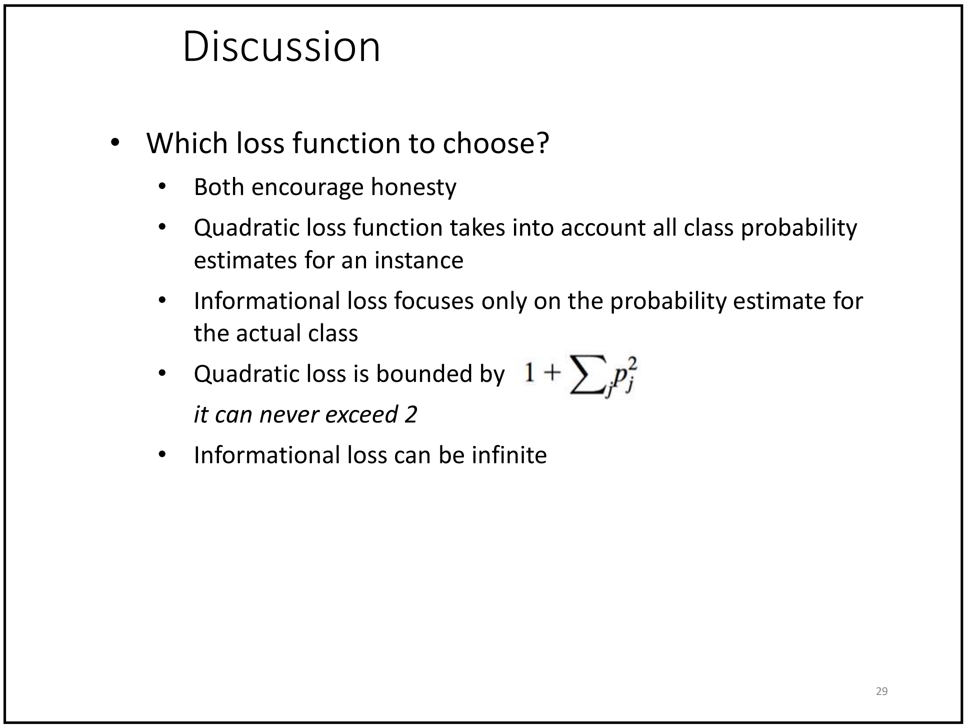
- The informational loss function is –log(p_c),
  where *c* is the index of the instance’s actual class
  - Number of bits required to communicate the actual class
- Let $p_1^* \dots p_k^*$ be the true class probabilities
- Then the expected value for the loss function is:

$$-p_1^* \log_2 p_1 - p_2^* \log_2 p_2 - \dots - p_k^* \log_2 p_k$$

- Justification for informational loss is that this is minimized when $p_j = p_j^*$:

$$-p_1^* \log_2 p_1^* - p_2^* \log_2 p_2^* - \dots - p_k^* \log_2 p_k^*$$

28

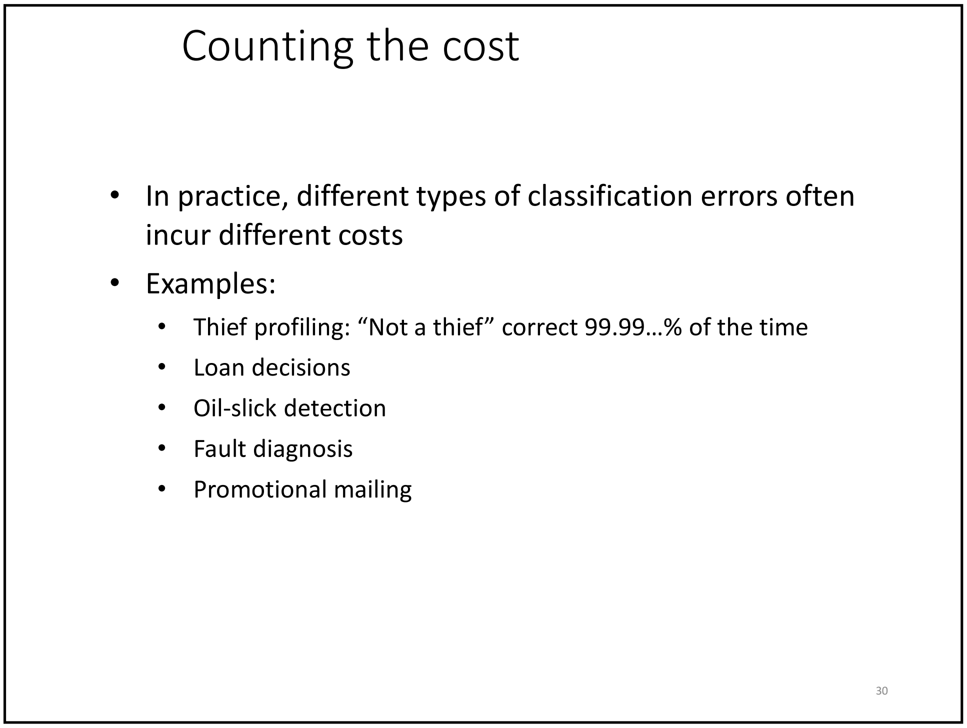


Discussion

- Which loss function to choose?
  - Both encourage honesty
  - Quadratic loss function takes into account all class probability estimates for an instance
  - Informational loss focuses only on the probability estimate for the actual class
  - Quadratic loss is bounded by $1+\sum_j p_j^2$
    it can never exceed 2
  - Informational loss can be infinite

29

29



Counting the cost

- In practice, different types of classification errors often incur different costs
- Examples:
  - Thief profiling: “Not a thief” correct 99.99…% of the time
  - Loan decisions
  - Oil-slick detection
  - Fault diagnosis
  - Promotional mailing

30

30

Counting the cost

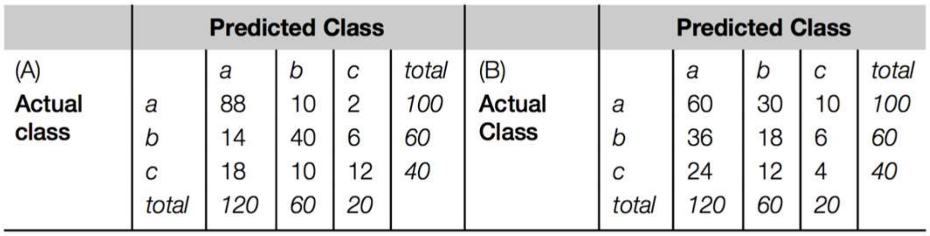
- The *confusion matrix*:

		Predicted class	
		Yes	No
Actual class	Yes	True positive	False negative
	No	False positive	True negative

- Different misclassification costs can be assigned to false positives and false negatives
- There are many other types of cost!
  - E.g., cost of collecting training data

Aside: the kappa statistic

- Two confusion matrices for a 3-class problem: actual predictor (left) vs. random predictor (right)

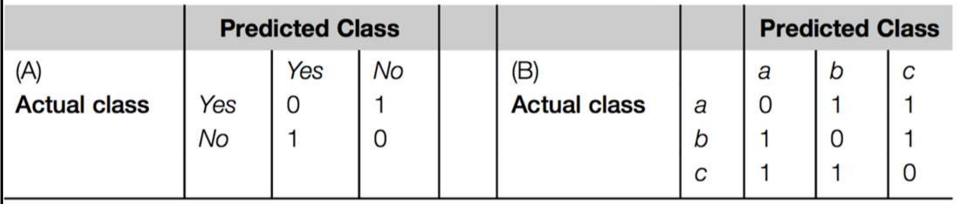


		Predicted Class						Predicted Class			
(A)		*a*	*b*	*c*	*total*	(B)		*a*	*b*	*c*	*total*
Actual	*a*	88	10	2	*100*	**Actual**	*a*	60	30	10	*100*
class	*b*	14	40	6	*60*	**Class**	*b*	36	18	6	*60*
	c	18	10	12	*40*		*c*	24	12	4	*40*
	total	*120*	*60*	*20*			*total*	*120*	*60*	*20*	

- Number of successes: sum of entries in diagonal (*D*)
- *Kappa* statistic: (success rate of actual predictor - success rate of random predictor) / (1 - success rate of random predictor)
- Measures relative improvement on random predictor: 1 means perfect accuracy, 0 means we are doing no better than random

Classification with costs

- Two cost matrices:



	Predicted Class						Predicted Class		
(A)		*Yes*	*No*		(B)		*a*	*b*	*c*
Actual class	*Yes*	0	1		**Actual class**	*a*	0	1	1
	No	1	0			*b*	1	0	1
						c	1	1	0

- In cost-sensitive evaluation of classification methods, success rate is replaced by average cost per prediction
  - Cost is given by appropriate entry in the cost matrix

Cost-sensitive classification

- Can take costs into account when making predictions
  - Basic idea: only predict high-cost class when very confident about prediction
- Given: predicted class probabilities
  - Normally, we just predict the most likely class
  - Here, we should make the prediction that minimizes the expected cost
    - Expected cost: dot product of vector of class probabilities and appropriate column in cost matrix
    - Choose column (class) that minimizes expected cost
- This is the minimum-expected cost approach to cost-sensitive classification

Lift metric in associative rules

The **lift** of a rule X-->Y is calculated as lift(X-->Y) = ((sup(X U Y)/ N) / (sup(X)/N *sup(Y)/ N), where

- N is the number of transactions in the transaction database,
- sup(X∪Y) is the number of transactions containing X and Y,
- sup(X) is the number of transactions containing X
- sup(Y) is the number of transactions containing Y.
- See the example of associative rule

```
rule 0:   4  ==> 2      support :  0.66 (4/6) confidence
:  1.0  lift :  1.0
rule 2:   1  ==> 5      support :  0.66 (4/6) confidence
:  1.0  lift :  1.2
rule 17:  1 4  ==> 2 5  support :  0.5 (3/6)   confidence
:  1.0  lift :  1.5
```

- For an association rule X ==> Y, if the lift is equal to 1, it means that X and Y are independent.
- If the lift is higher than 1, it means that X and Y are positively correlated.
- If the lift is lower than 1, it means that X and Y are negatively correlated.
- For example, if we consider the rule {1, 4} ==> {2, 5}, it has a lift of 1.5, which means that the occurrence of the itemset {1, 4} is positively correlated with the occurrence of {2, 5}.

35

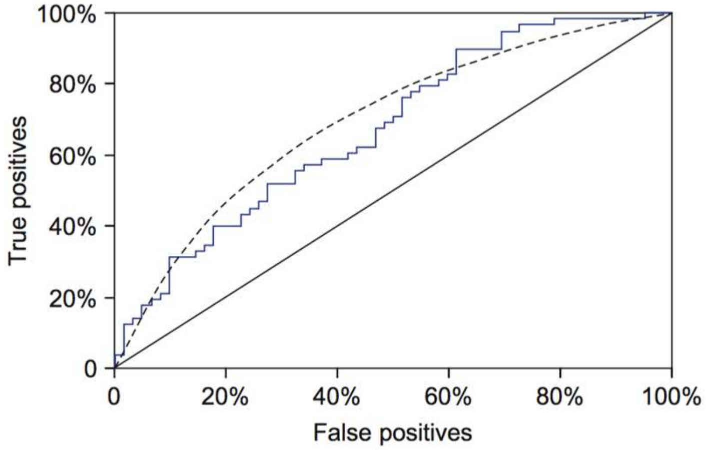
ROC curves

- *ROC curves* are similar to lift charts
  - Stands for "receiver operating characteristic"
  - Used in signal detection to show tradeoff between hit rate and false alarm rate over noisy channel
- Differences to lift chart:
  - *y* axis shows percentage of true positives in sample *rather than absolute number*
  - *x* axis shows percentage of false positives in sample *rather than sample size*

36

36

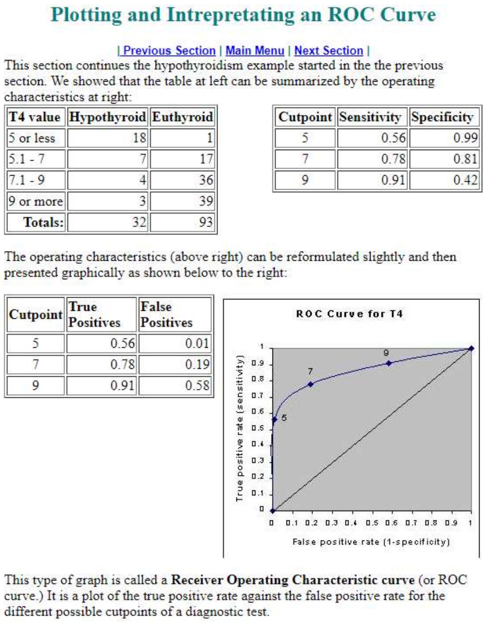
A sample ROC curve



- Jagged curve—one set of test data
- Smoother curve—use cross-validation

A sample ROC curve

Illustrative example:
http://gim.unmc.edu/dxtests/roc2.htm



Plotting and Intrepretating an ROC Curve

This section continues the hypothyroidism example started in the the previous section. We showed that the table at left can be summarized by the operating characteristics at right:

T4 value	Hypothyroid	Euthyroid
5 or less	18	1
5.1 - 7	7	17
7.1 - 9	4	36
9 or more	3	39
Totals:	32	93

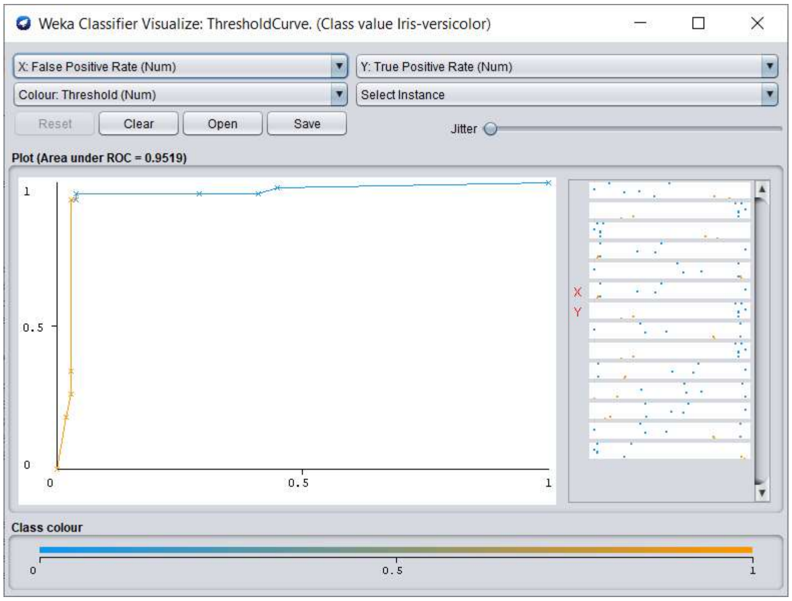
Cutpoint	Sensitivity	Specificity
5	0.56	0.99
7	0.78	0.81
9	0.91	0.42

The operating characteristics (above right) can be reformulated slightly and then presented graphically as shown below to the right:

Cutpoint	True Positives	False Positives
5	0.56	0.01
7	0.78	0.19
9	0.91	0.58

This type of graph is called a **Receiver Operating Characteristic curve** (or ROC curve.) It is a plot of the true positive rate against the false positive rate for the different possible cutpoints of a diagnostic test.

A sample ROC curve – visualize threshold curve



Weka Classifier Visualize: ThresholdCurve. (Class value Iris-versicolor)

X: False Positive Rate (Num)

Y: True Positive Rate (Num)

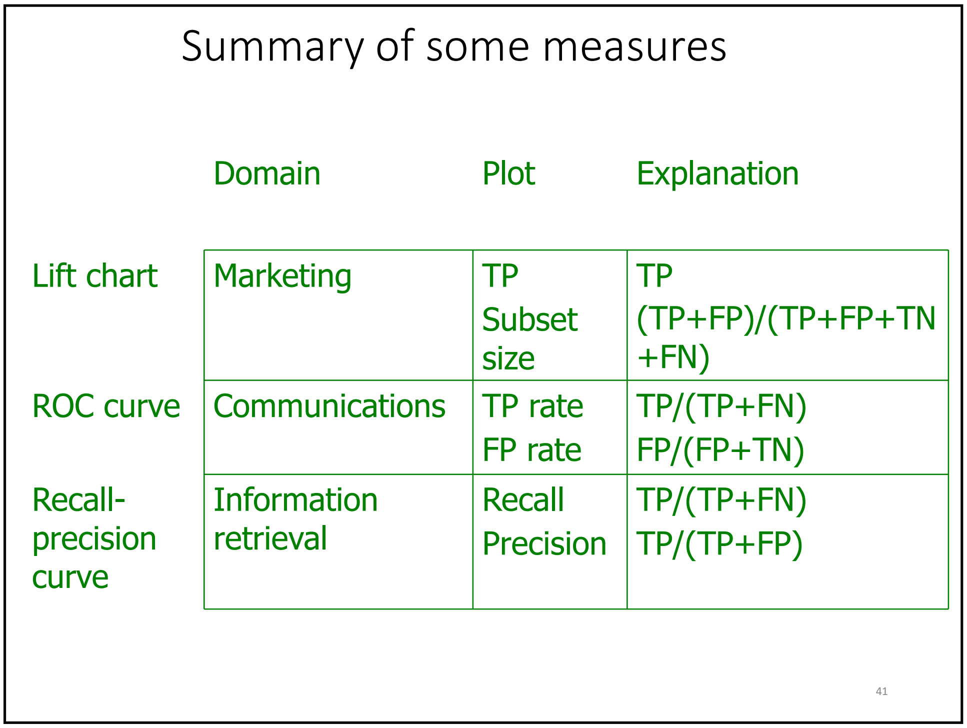
Colour: Threshold (Num)

Select Instance

Plot (Area under ROC = 0.9519)

More measures...

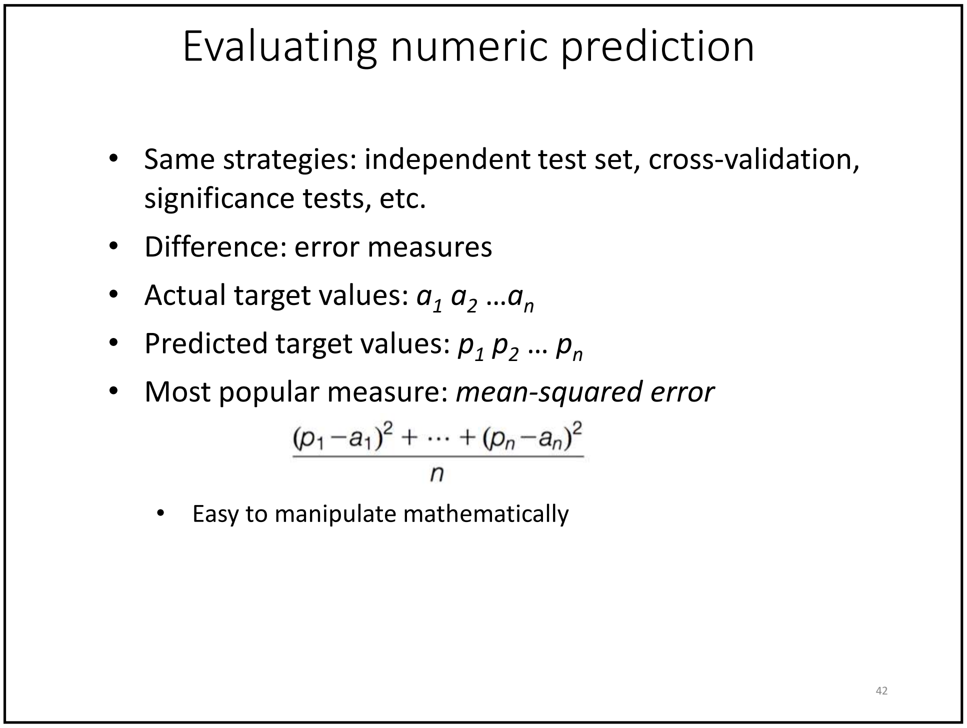
- Percentage of retrieved documents that are relevant: *precision*=TP/(TP+FP)
- Percentage of relevant documents that are returned: *recall* =TP/(TP+FN)
- Precision/recall curves have hyperbolic shape
- Summary measures: average precision at 20%, 50% and 80% recall (*three-point average recall*)
- *F-measure*=(2 × recall × precision)/(recall+precision)
- *sensitivity* = TP / (TP + FN)
- specificity = TN / (FP + TN)
- Area under the ROC curve (*AUC*):
  measure of how well a parameter can distinguish between two diagnostic groups (diseased/normal)



Summary of some measures

	Domain	Plot	Explanation
Lift chart	Marketing	TP Subset size	TP (TP+FP)/(TP+FP+TN+FN)
ROC curve	Communications	TP rate FP rate	TP/(TP+FN) FP/(FP+TN)
Recall-precision curve	Information retrieval	Recall Precision	TP/(TP+FN) TP/(TP+FP)

41



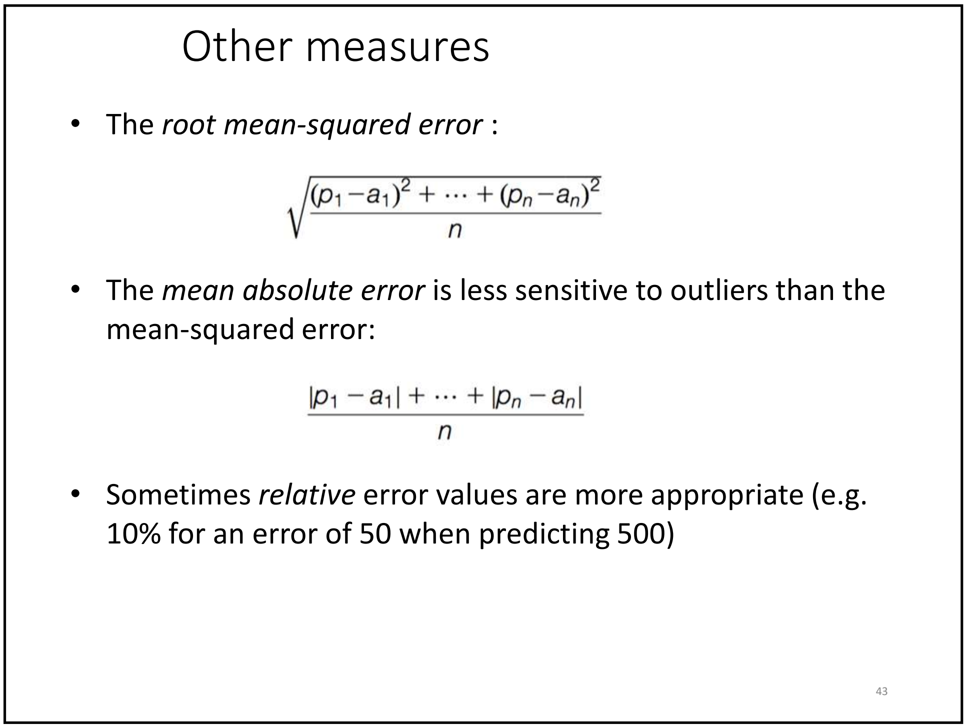
Evaluating numeric prediction

- Same strategies: independent test set, cross-validation, significance tests, etc.
- Difference: error measures
- Actual target values: $a_1\ a_2 \ldots a_n$
- Predicted target values: $p_1\ p_2 \ldots\ p_n$
- Most popular measure: *mean-squared error*

$$\frac{(p_1-a_1)^2+\cdots+(p_n-a_n)^2}{n}$$

  - Easy to manipulate mathematically

42



Other measures

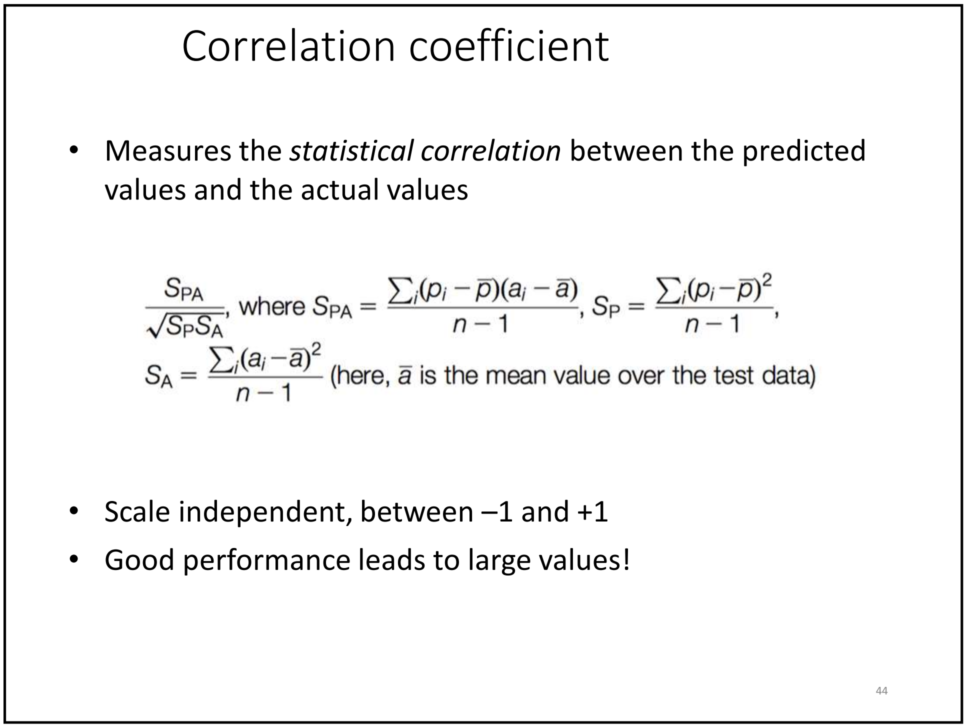
- The *root mean-squared error* :

$$\sqrt{\frac{(p_1-a_1)^2+\cdots+(p_n-a_n)^2}{n}}$$

- The *mean absolute error* is less sensitive to outliers than the mean-squared error:

$$\frac{|p_1-a_1|+\cdots+|p_n-a_n|}{n}$$

- Sometimes *relative* error values are more appropriate (e.g. 10% for an error of 50 when predicting 500)



Correlation coefficient

- Measures the *statistical correlation* between the predicted values and the actual values

$$\frac{S_{\mathrm{PA}}}{\sqrt{S_{\mathrm{P}}S_{\mathrm{A}}}}, \text{ where } S_{\mathrm{PA}}=\frac{\sum_i(p_i-\overline{p})(a_i-\overline{a})}{n-1},\ S_{\mathrm{P}}=\frac{\sum_i(p_i-\overline{p})^2}{n-1},$$

$$S_{\mathrm{A}}=\frac{\sum_i(a_i-\overline{a})^2}{n-1} \text{ (here, } \overline{a} \text{ is the mean value over the test data)}$$

- Scale independent, between –1 and +1
- Good performance leads to large values!

Which measure?

- Best to look at all of them
- Often it doesn't matter
- Example:

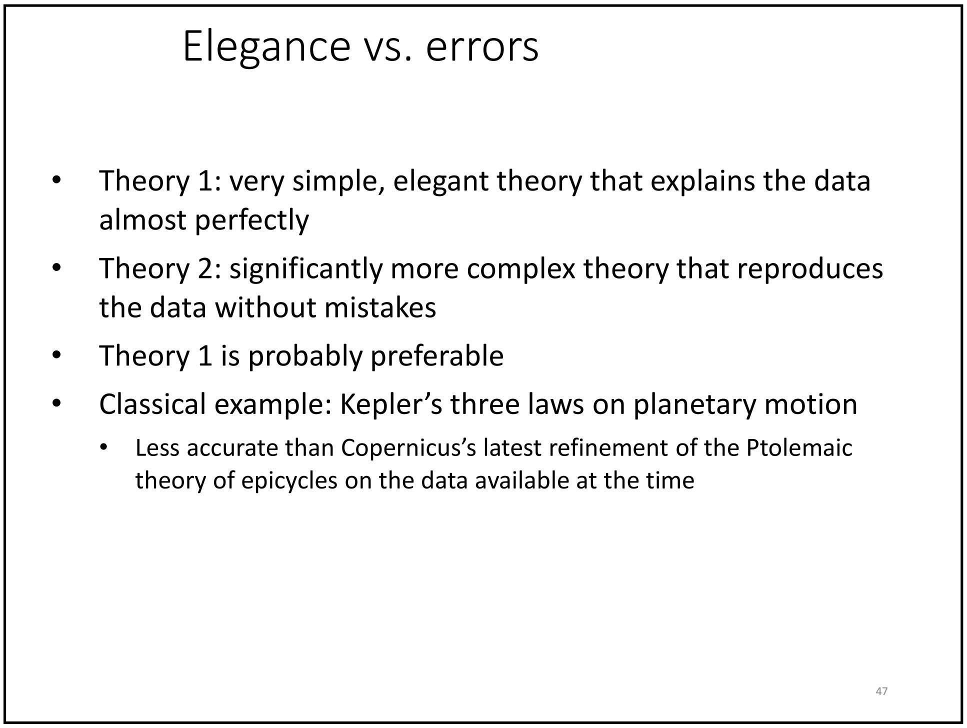
	A	B	C	D
Root mean-squared error	67.8	91.7	63.3	57.4
Mean absolute error	41.3	38.5	33.4	29.2
Root rel squared error	42.2%	57.2%	39.4%	35.8%
Relative absolute error	43.1%	40.1%	34.8%	30.4%
Correlation coefficient	0.88	0.88	0.89	0.91

- D best
- C second-best
- A, B arguable

Model selection criteria

- Model selection criteria attempt to find a good compromise between:
  - The complexity of a model
  - Its prediction accuracy on the training data
- Reasoning: a good model is a simple model that achieves high accuracy on the given data
- Also known as *Occam's Razor* : the best theory is the smallest one that describes all the facts

William of Ockham, born in the village of Ockham in Surrey (England) about 1285, was the most influential philosopher of the 14th century and a controversial theologian.



Elegance vs. errors

- Theory 1: very simple, elegant theory that explains the data almost perfectly
- Theory 2: significantly more complex theory that reproduces the data without mistakes
- Theory 1 is probably preferable
- Classical example: Kepler’s three laws on planetary motion
  - Less accurate than Copernicus’s latest refinement of the Ptolemaic theory of epicycles on the data available at the time