Machine Learning Applications

Evaluierung

- Hold-out Estimates
- Cross-validation
- Significance Testing
- Sign test

Basierende auf Folien von Johannes Fürnkranz.
Danke fürs Offenlegen der Folien

Evaluation of Learned Models

- **Validation through experts**
  - a domain expert evaluates the plausibility of a learned model

    + but often the only option (e.g., clustering)

    - subjective, time-intensive, costly
- **Validation on data**
  - evaluate the accuracy of the model on a separate dataset drawn from the same distribution as the training data

    - labeled data are scarce, could be better used for training

    + fast and simple, off-line, no domain knowledge needed, methods for re-using training data exist (e.g., cross-validation)
- **On-line Validation**
  - test the learned model in a fielded application

    + gives the best estimate for the overall utility

    - bad models may be costly

Confusion Matrix
(Classification)

	Classified as +	Classified as –	
Is +	true positives (tp)	false negatives (fn)	tp + fn = P
Is –	false positives (fp)	true negatives (tn)	fp + tn = N
	tp + fp	fn + tn	|E| =P + N

- the confusion matrix summarizes all important information
- how often is class *i* confused with class *j*
- most evaluation measures can be computed from the confusion matrix
- accuracy
- recall/precision, sensitivity/specificity
- ...

Basic Evaluation Measures

- true positive rate: $$tpr = \frac{tp}{tp + fn}$$
  - percentage of *correctly* classified *positive* examples
- false positive rate: $$fpr = \frac{fp}{fp + tn}$$
  - percentage of negative examples *incorrectly* classified *as positive*
- false negative rate: $$fnr = \frac{fn}{tp + fn} = 1 - tpr$$
  - percentage of positive examples *incorrectly* classified *as negative*
- true negative rate: $$tnr = \frac{tn}{fp + tn} = 1 - fpr$$
  - percentage of *correctly* classified *negative* examples
- accuracy: $$acc = \frac{tp + tn}{P + N}$$
  - percentage of correctly classified examples
- can be written in terms of *tpr* and *fpr*: $$acc = \frac{P}{P + N} \cdot tpr + \frac{N}{P + N} \cdot (1 - fpr)$$
- error: $$err = \frac{fp + fn}{P + N} = 1 - acc = \frac{P}{P + N} \cdot (1 - tpr) + \frac{N}{P + N} \cdot fpr$$
  - percentage of incorrectly classified examples

Confusion Matrix (Multi-Class Problems)

- for multi-class problems, the confusion matrix has many more entries:

classified as

true class	A	B	C	D	
A	$n_{A,A}$	$n_{B,A}$	$n_{C,A}$	$n_{D,A}$	n_A
B	$n_{A,B}$	$n_{B,B}$	$n_{C,B}$	$n_{D,B}$	n_B
C	$n_{A,C}$	$n_{B,C}$	$n_{C,C}$	$n_{D,C}$	n_C
D	$n_{A,D}$	$n_{B,D}$	$n_{C,D}$	$n_{D,D}$	n_D
	$\overline{n}_A$	$\overline{n}_B$	$\overline{n}_C$	$\overline{n}_D$	$\|E\|$

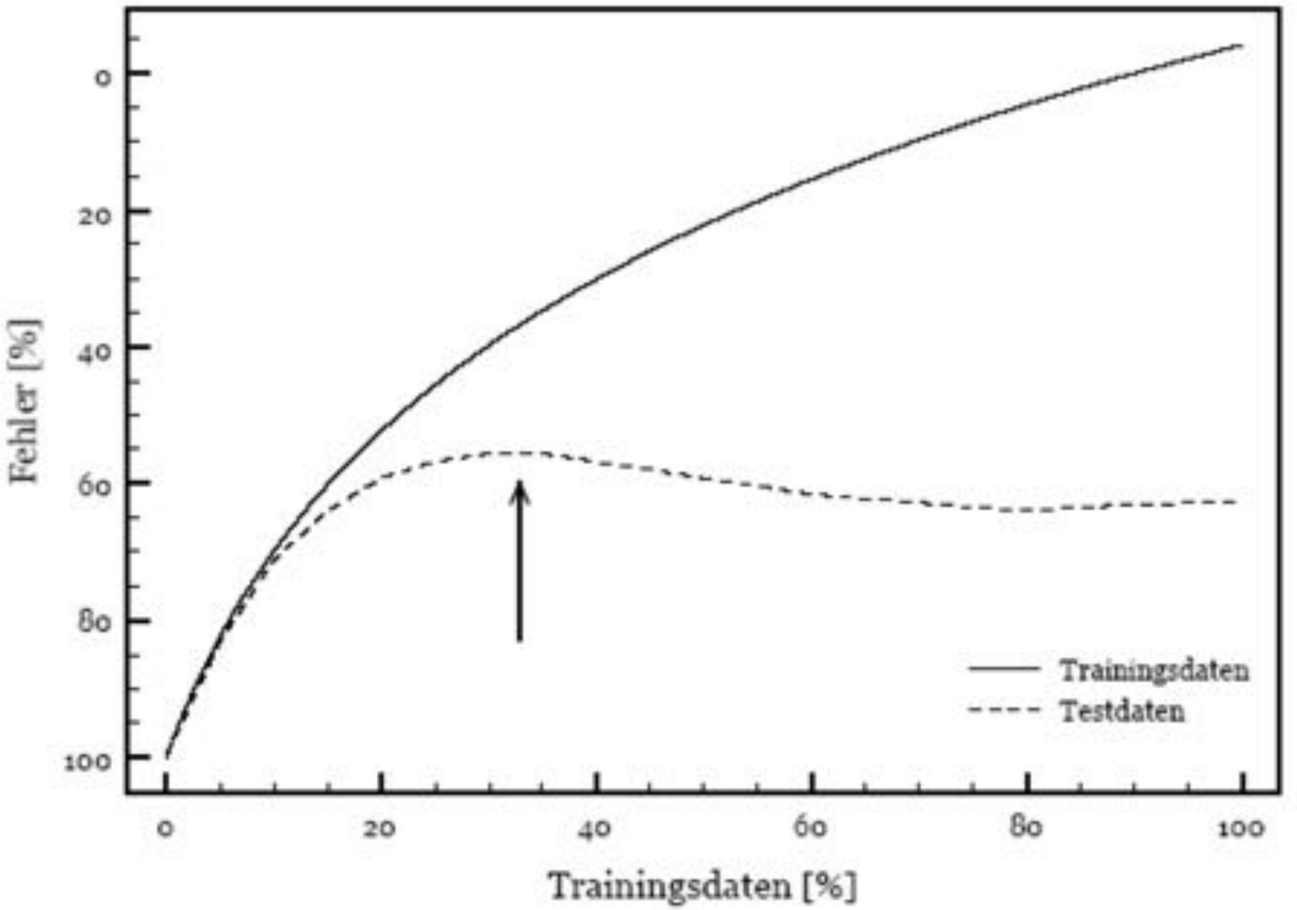
- accuracy is defined analogously to the two-class case:

$$accuracy = \frac{n_{A,A} + n_{B,B} + n_{C,C} + n_{D,D}}{|E|}$$

Out-of-Sample Testing

- **Performance cannot be measured on training data**
  - overfitting!
- **Reserve a portion of the available data for testing**
  - typical scenario
    - 2/3 of data for training
    - 1/3 of data for testing (evaluation)
  - a classifier is trained on the training data
  - and tested on the test data
    - e.g., confusion matrix is computed for test data set
- **Problems:**
  - waste of data
  - labelling may be expensive
  - high variance
    - often: repeat 10 times or → cross-validation

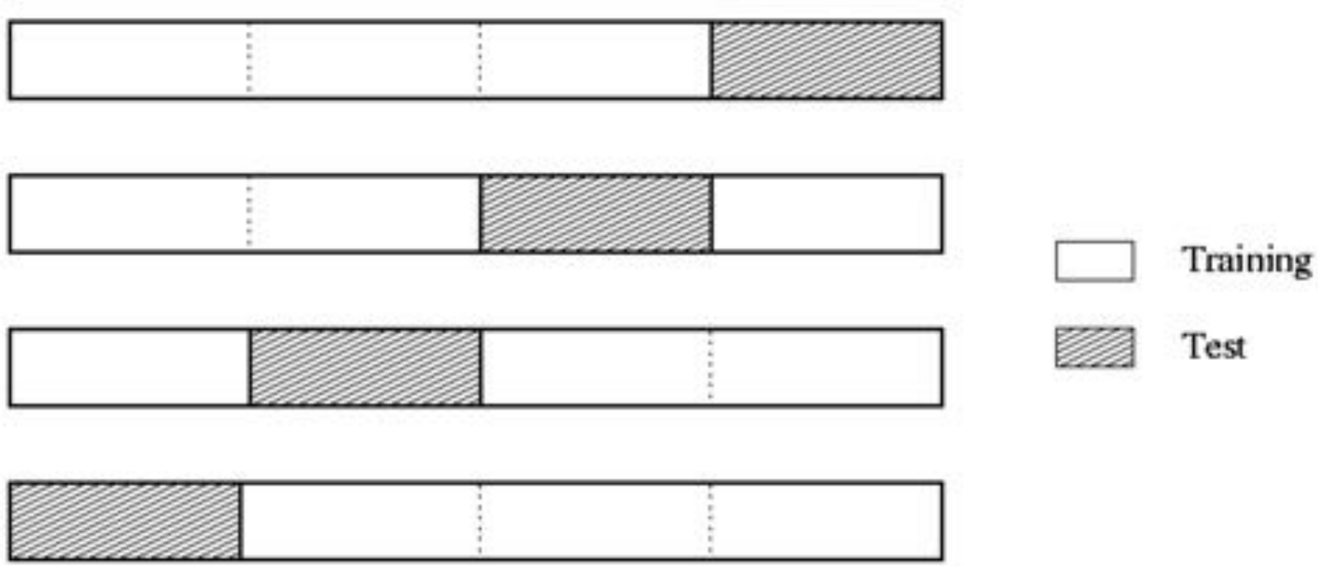
Typical Learning Curves



Quelle: Winkler 2007, nach Mitchell 1997,

Cross-Validation

- Algorithm:
  - split dataset into *x* (usually 10) partitions
  - for every partition *X*
    - use other *x*-1 partitions for learning and partition *X* for testing
  - average the results
- Example: 4-fold cross-validation



Leave-One-Out Cross-Validation

- ***n*-fold cross-validation**
  - where *n* is the number of examples:
    - use *n*-1 examples for training
    - 1 example for testing
    - repeat for each example
- **Properties:**

  \+ makes best use of data
  - only one example not used for testing

  \+ no influence of random sampling
  - training/test splits are determined deterministically

  – typically very expensive
  - but, e.g., not for k-NN (Why?)

  – bias
  - e.g., majority classifier in a perfectly balanced problem

Experimental Evaluation of Algorithms

- Typical experimental setup (in % Accuracy):
  - evaluate *n* algorithms on *m* datasets

Dataset	Grading	Select	Stacking	Voting
audiology	83.36	77.61	76.02	84.56
autos	80.93	80.83	82.20	83.51
balance-scale	89.89	91.54	89.50	86.16
breast-cancer	73.99	71.64	72.06	74.86
breast-w	96.70	97.47	97.41	96.82
colic	84.38	84.48	84.78	85.08
credit-a	86.01	84.87	86.09	86.04
credit-g	75.64	75.48	76.17	75.23
diabetes	75.53	76.86	76.32	76.25
glass	74.35	74.44	76.45	75.70
heart-c	82.74	84.09	84.26	81.55
heart-h	83.64	85.78	85.14	83.16
heart-statlog	84.22	83.56	84.04	83.30

Dataset	Grading	Select	Stacking	Voting
hepatitis	83.42	83.03	83.29	82.77
ionosphere	91.85	91.34	92.82	92.42
iris	95.13	95.20	94.93	94.93
labor	93.68	90.35	91.58	93.86
lymph	83.45	81.69	80.20	84.05
primary-t.	49.47	49.23	42.63	46.02
segment	98.03	97.05	98.08	98.14
sonar	85.05	85.05	85.58	84.23
soybean	93.91	93.69	92.90	93.84
vehicle	74.46	73.90	79.89	72.91
vote	95.93	95.95	96.32	95.33
vowel	98.74	99.06	99.00	98.80
zoo	96.44	95.05	93.96	97.23

- Can we conclude that algorithm X is better than Y? How?

Summarizing Experimental Results

- Averaging the performance
  - May be deceptive:
    - algorithm A is 0.1% better on 19 datasets with thousands of examples
    - algorithm B is 2% better on 1 dataset with 50 examples
    - A is better, but B has the higher average accuracy
  - In our example: “Grading” is best on average

Dataset	Grading	Select	Stacking	Voting
Avg	85.04	84.59	84.68	84.88

- Counting wins/ties/losses
  - now “Stacking” is best
  - Results are “inconsistent”:
    - Gradsing > Select > Voting > Grading
  - How many “wins” are needed to conclude that one method is better than the other?

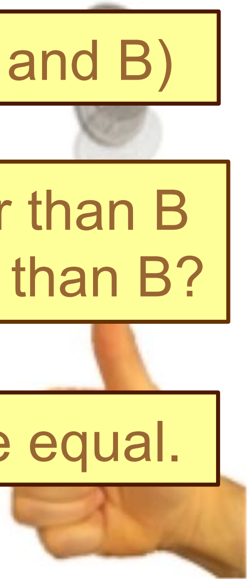
	Grading	Select	Stacking	Voting
Grading	—	15/1/10	11/0/15	12/0/14
Select	10/1/15	—	10/0/16	14/0/12
Stacking	15/0/11	16/0/10	—	15/1/10
Voting	14/0/12	12/0/14	10/1/15	—

Sign Test

- Given:
  - A coin with two sides (heads and tails)
- Question:
  - How often do we need heads in order to be sure that the coin is not fair?
- Null Hypothesis:
  - The coin is fair (P(heads) = P(tails) = 0.5)
  - We want to refute that!
- Experiment:
  - Throw up the coin N times
- Result:
  - i heads, $N- i$ tails
  - What is the probability of observing i under the null hypothesis?

Sign Test

- Given:
  - ~~A coin with two sides (heads and~~ Two Learning Algorithms (A and B)
- Question:
  - ~~How often do we ne~~ ~~the coin is not fair?~~ On how many datasets must A be better than B to ensure that A is a better algorithm than B?
- Null Hypothesis:
  - ~~The coin is fair (P(heads) = P(tails) = 0.5)~~ Both Algorithms are equal.
  - We want to refute that!
- Experiment:
  - ~~Throw up the coin *N* times~~ Run both algorithms on N datasets
- Result:
  - ~~*i* heads, *N*- *i* tails~~ i wins for A on N-i wins for B
  - What is the probability of observing *i* under the null hypothesis?



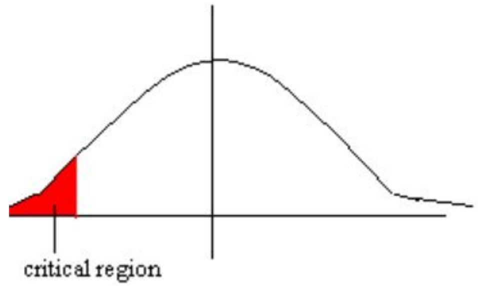
Sign Test: Summary

We have a binomial distribution with p = ½

- the probability of having i successes is $P(i) = \binom{N}{i} p^i (1-p)^{N-i}$

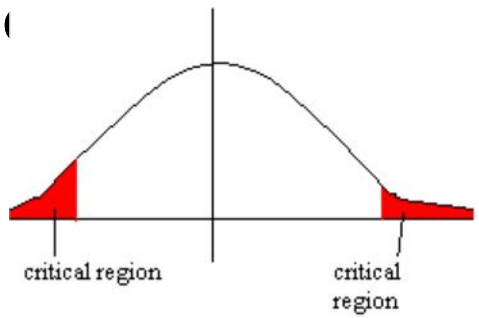
- the probability of having at most k successes is (one-tailed test)

$$P(i \leq k) = \sum_{i=1}^{k} \binom{N}{i} \frac{1}{2^i} \cdot \frac{1}{2^{N-i}} = \frac{1}{2^N} \sum_{i=1}^{k} \binom{N}{i}$$



- the probability of having at most k successes or at l
N- k successes is (two-tailed test)

$$P(i \leq k \vee i \geq N-k) = \frac{1}{2^N} \sum_{i=1}^{k} \binom{N}{i} + \frac{1}{2^N} \sum_{i=1}^{k} \binom{N}{N-i} = \frac{1}{2^{N-1}} \sum_{i=1}^{k} \binom{N}{i}$$



- for large N, this can be approximated with a normal distribution

Illustrations taken from http://www.mathsrevision.net/

Source: Erich Mittenecker, *Planung und Statistische Auswertung von Experimenten*, 10th edition, Deuticke Verlag, Wien, 1983.

Table Sign Test

- Example:
  - 20 datasets
  - Alg. A vs. B
    - A 4 wins
    - B 14 wins
    - 2 ties (not counted)
  - we can say with a certainty of 95% that B is better than A
  - but not with 99% certainty!

Vorzeichentest: Kritische Häufigkeiten i bzw. $N-i$ (s. S. 167)

N	Irrtumswahrscheinlichkeit 1%	5%	N	Irrtumswahrscheinlichkeit 1%	5%
6	—	0	41	11	13
7	—	0	42	12	14
8	0	0	43	12	14
9	0	1	44	13	15
10	0	1	45	13	15
11	0	1	46	13	15
12	1	2	47	14	16
13	1	2	48	14	16
14	1	2	49	15	17
15	2	3	50	15	17
16	2	3	51	15	18
17	2	4	52	16	18
18	3	4	53	16	18
19	3	4	54	17	19
20	3	5	55	17	19
21	4	5	56	17	20
22	4	5	57	18	20
23	4	6	58	18	21
24	5	6	59	19	21
25	5	7	60	19	21
26	6	7	61	20	22
27	6	7	62	20	22
28	6	8	63	20	23
29	7	8	64	21	23
30	7	9	65	21	24
31	7	9	66	22	24
32	8	9	67	22	25
33	8	10	68	22	25
34	9	10	69	23	25
35	9	11	70	23	26
36	9	11	71	24	26
37	10	12	72	24	27

- Online: http://www.fon.hum.uva.nl/Service/Statistics/Sign_Test.html

Properties

- **Sign test is a very simple test**
  - does not make any assumption about the distribution
- **Sign test is very conservative**
  - If it detects a significant difference, you can be sure it is
  - If it does not detect a significant difference, a different test that models the distribution of the data may still yield significance
- **Alternative tests:**
  - two-tailed *t*-test:
    - incorporates magnitude of the differences in each experiment
    - assumes that differences form a normal distribution
- **Rule of thumb:**
  - Sign test answers the question “How often?”
  - t-test answers the question “How much?”

Problem of Multiple Comparisons

- **Problem:**
  - With 95% certainty we have
    - a probability of 5% that one algorithm appears to be better than the other
    - even if the null hypothesis holds!

    → if we make many pairwise comparisons the chance that a “significant” difference is observed increases rapidly

- **Solutions:**
  - Bonferroni adjustments:
    - **Basic idea:** tighten the significance thresholds depending on the number of comparisons
    - Too conservative
  - Friedman and Nemenyi tests
    - recommended procedure (based on average ranks)

    → Demsar, *Journal of Machine Learning Research* 7, 2006
    http://jmlr.csail.mit.edu/papers/v7/demsar06a.html