CSE517A Machine Learning **Spring 2024**

Lecture A3: Performance Evaluation of ML Methods

Instructor: Marion Neumann

Reading: FCML: 5.4 (Performance); ESL: 7.10 (Cross-Validation);

1 Introduction

Comparing different machine learning methods or model choices such as using different hyperparameters, features, kernels, etc. is an important step in ML research and in the development of ML approaches in real-world applications. Typically our goal is to find the *best* method. To do so we need to be able to measure the *performance* of each approach/model/parameter setting etc. and compare those measures.

What does the **performance of a ML method** depend on?

- choice of learning algorithm/model
- tuning of model (parameters/hyper-parameters)
- training dataset D and test dataset
- performance measure
- statistical test

What exactly is the **purpose** of performance evaluation?

(1) ML research: compare new algorithm/method to others
   - specific domain
   - set of benchmark domains (general effectiveness)

(2) ML applications: compare multiple algorithm/methods
   - specific domain (find the best algorithm for a given application)
   - set of benchmark domains

1.1 Example Performance Evaluation Scenarios

Compare different models for a *given application* with the goal to find the **best model** for this particular application.

The example scenario on the left is for a *classification task* (breast cancer).

Algo	Acc	RMSE	TPR	FPR	Prec	Rec	F	AUC
NB	71.7	.4534	.44	.16	.53	.44	.48	.7
C4.5	75.5	.4324	.27	.04	.74	.27	.4	.59
3NN	72.4	.5101	.32	.1	.56	.32	.41	.63
Ripp	71	.4494	.37	.14	.52	.37	.43	.6
SVM	69.6	.5515	.33	.15	.48	.33	.39	.59
Bagg	67.8	.4518	.17	.1	.4	.17	.23	.63
Boost	70.3	.4329	.42	.18	.5	.42	.46	.7
RanF	69.23	.47	.33	.15	.48	.33	.39	.63

Evaluating Learning Algorithms: A Classification Perspective
Nathalie Japkowicz & Mohak Shah Cambridge University Press, 2011

[1] For further reading – especially for PhD students – I recommend this book: Evaluating Learning Algorithms: A Classification Perspective by Nathalie Japkowicz and Mohak Shah (Cambridge University Press, 2011).

Compare different models for a *given application* with the goal to show that your **model** is **better** for this application.

The example scenario on the left is for a *regression task* (traffic prediction).

City	Model	Public Transport R^2	Public Transport $RMSE$	Public Transport MAE	Public Transport NLPD	Pedestrian R^2	Pedestrian $RMSE$	Pedestrian MAE	Pedestrian NLPD
FR									
	kNN	0.75	232	112	—	0.63	210	76	—
	GP	0.75	233	108	-0.73	0.68 ◇	188	78	-1.78
	SGP	0.76	232	108	-0.73	0.70 •	180	77	-1.81 •
	XGP	0.76	228	107	-0.74	0.71 •	171 •	75 •	-1.23 •
	SXGP	0.77 ⋆	224	108	-1.10 ⋆	0.74 ⋆	163 ⋆	73	-1.12
KL									
	kNN	0.58	32	22	—	0.50	53	27	—
	GP	0.80 ◇	22 ◇	14 ◇	-1.08	0.51 •	51 •	28 •	-0.20
	SGP	0.81 •	21 •	14	-1.10 •	0.55 •	49 •	27 •	-0.38
	XGP	0.84 •	19 •	13	-1.07	0.57	48	27	-0.33
	SXGP	0.85 ⋆	19 ⋆	13 ⋆	-1.05	0.60 ⋆	48 ⋆	26 ⋆	-0.56

Table 4.2: Results of the models kNN, GP, SGP, XGP, SXGP for 2 German cities. The evaluation is based on the coefficient of determination R^2, the error measures $RMSE$ and MAE as well as the negative log predictive density $NLPD$. The following notation is used for indicating significant improvements: ◇ GP improves kNN; • SGP/ XGP improve GP; ⋆ SXGP improves XGP.

Compare different features for a *given application* and ML method with the goal to find the **best features** for this model and application.

The example scenario on the left is for a *classification task* (plant disease classification).

Table 4. Average accuracies and standard errors in % (average prediction time in seconds) of the erosion feature ensemble (EFE), all features (ALL), most frequent class (MFC) and random (RAND) for disease detection (DETECT) and classification (CLASSIFY)

		EFE	ALL	MFC	RAND
DETECT	STUDY	93.9 ± 1.9 (0.00″)	92.5 ± 1.6 (0.00″)	81.3	50.0
	FULL	93.3 ± 0.4 (0.02″)	95.4 ± 0.5 (0.15″)	62.6	50.0
CLASSIFY	STUDY	75.2 ± 1.9 (0.00″)	77.0 ± 2.6 (0.01″)	19.3	16.7
	FULL	83.8 ± 0.7 (0.04″)	87.6 ± 0.6 (0.38″)	37.4	16.7

Compare different kernels for *different applications* and a specific ML method with the goal to find the **best kernel** or show that your novel kernel is effective.

The example scenario on the left is for various *classification tasks*.

Table 4.5: **Accuracies on Attributed Graphs.** Average accuracies ± standard error of 10-fold cross validation (10 runs). The kernel parameters t_{MAX} for all PKS and h_{MAX} for WL were learned on the training splits ($t_{MAX}, h_{MAX} \in \{0, 1, \ldots, 10\}$). Whenever the normalized version of a kernel performed better than the unnormalized version we report these results and mark the method with $*$. CSM is implemented in Java and computations were performed on a machine with 32 GB of memory. OUT OF MEMORY indicates a Java *outOfMemeoryError*. Bold indicates that the method performs significantly better than the second best method under a paired t-test ($p < 0.05$). The SVM cost parameter is learned on the training splits. We choose $c \in \{10^{-7}, 10^{-5}, \ldots, 10^{5}, 10^{7}\}$ for normalized kernels and $c \in \{10^{-7}, 10^{-5}, 10^{-3}, 10^{-1}\}$ for unnormalized kernels.

method	dataset SYNTHETIC	ENZYMES	PROTEINS	PRO-FULL	BZR	COX2	DHFR
PK	99.2 ± 0.1*	65.9 ± 0.4*	76.3 ± 0.2*	75.7 ± 0.4*	88.1 ± 0.2*	79.4 ± 0.6	84.1 ± 0.3*
P2K	98.7 ± 0.1	68.1 ± 0.5	75.9 ± 0.2*	**76.9** ± 0.2*	88.8 ± 0.2	80.9 ± 0.4	83.5 ± 0.3*
SP	99.0 ± 0.1	64.3 ± 0.3*	73.2 ± 0.2*	59.9 ± 0.0*	85.2 ± 0.2*	78.5 ± 0.1	79.7 ± 0.2*
GH	50.0 ± 0.0*	71.2 ± 0.2*	73.0 ± 0.1*	60.9 ± 0.0*	84.8 ± 0.4	79.5 ± 0.2*	80.0 ± 0.2
CSM	99.0 ± 0.1	**72.8** ± 0.4*	OUT OF MEMORY	OUT OF MEMORY	87.0 ± 0.2*	79.2 ± 0.4*	80.1 ± 0.3*

1.2 Performing Scientific Experiments

Performance comparisons are typically based on results from *cross-validation* or *re-sampling*. To summarize the results we typically only present aggregated/average statistics such as the *average accuracies* and hopefully their *standard deviations* or *standard errors*. To make the comparison stronger we also include the result of a *test to assess statistical significance*. These results only make sense if we use exactly the **same** training and test sets/cross-validation splits for *all* compared methods/settings.
Comparing aggregated or previously reported measures can only give a *vague hint* on which method performs better – there is **little to no scientific value** in such comparisons.

Performing scientific experiments is key!

2 Assessing Performance

2.1 Performance Measures for Regression

2.1.1 Error

Mean absolute error (MAE):

$$e = \frac{1}{n}\sum_{i=1}^{n}|f(\mathbf{x}_i) - y_i| \tag{1}$$

Mean squared error (MSE):

$$e = \frac{1}{n}\sum_{i=1}^{n}(f(\mathbf{x}_i) - y_i)^2 \tag{2}$$

2.1.2 Coefficient of determination

Coefficient of determination is used to measure how well the regression function approximates the observed data.

Define

$$\begin{aligned} R^2 &= 1 - \frac{SS_{RES}}{SS_{TOT}} \\ &= \frac{SS_{REG}}{SS_{TOT}} \\ &= \frac{\frac{1}{n}SS_{REG}}{\frac{1}{n}SS_{TOT}} \end{aligned} \tag{3}$$

where

$$SS_{RES} = \sum_{i=1}^{n}(f(\mathbf{x}_i) - y_i)^2 \tag{4}$$

$$SS_{REG} = \sum_{i=1}^{n}(f(\mathbf{x}_i) - \bar{y})^2 \tag{5}$$

$$SS_{TOT} = \sum_{i=1}^{n}(y_i - \bar{y})^2 \tag{6}$$

For linear regression, we have $SS_{RES} + SS_{REG} = SS_{TOT}$. $\frac{1}{n}SS_{RES}$ is the variance of the residuals, $\frac{1}{n}SS_{REG}$ is the variance of the model's predictions and $\frac{1}{n}SS_{TOT}$ is the sample variance.

2.1.3 Negative log predictive density

The smaller negative log predictive density is, the model is better.

$$NLPD = -\log p(y) \tag{7}$$

$$= -\frac{1}{n}\sum_{i=1}^{n}\log p(y_i = f(\mathbf{x}_i) \mid \mathbf{x}_i) \tag{8}$$

For example, for Gaussian Process, $NLPD = \frac{1}{n}\sum_{i=1}^{n}(\frac{1}{2}\log(2\pi\sigma_i^2) + \frac{(y_i-\mu_i)^2}{2\sigma_i^2})$.

2.2 Performance Measures for Classification

2.2.1 Error and Accuracy

The most simple measure for a classifier's performance is the 0/1 loss:

$$\frac{1}{n}\sum_{i=1}^{n}\delta_{h(\mathbf{x}_i)\neq y_i} \times 100\% \tag{9}$$

Equivalently, we can report the accuracy, which is just 1-error (×100%):

$$\frac{1}{n}\sum_{i=1}^{n}\delta_{h(\mathbf{x}_i)=y_i} \times 100\% \tag{10}$$

Note: when using these measures, false positives and false negatives are treated equally.

2.2.2 Confusion Matrix

Application

Scientists working on real-world applications such as plant disease classification using cell-phone images are interested in which class is mistaken for which. Presenting and interpreting the confusion matrix is helpful in such cases:

For full the confusion matrix to analyze the per class performance of EFE is

$$M = \begin{pmatrix} 1009 & 26 & 2 & 58 & 15 & 6 \\ 39 & 891 & 0 & 94 & 28 & 14 \\ 2 & 0 & 65 & 1 & 0 & 0 \\ 44 & 64 & 5 & 335 & 17 & 3 \\ 9 & 24 & 0 & 6 & 163 & 16 \\ 2 & 1 & 0 & 0 & 2 & 16 \end{pmatrix}$$

where the classes are *n-inf*, *cerc*, *rust*, *pseu*, *ram*, and *phom* (top to bottom resp. left to right), and the off-diagonals of the rows (columns) show the number of false positives (false negatives). All classes except *phom* have classification accuracies above 67%. The misclassified leaf spots caused by Phoma were frequently labeled as Cercospora or Ramularia. This is due to their similar appearance and the lack of sufficient training data in this class; less than 2% of the considered regions have the label *phom*. These results suggest that we can achieve even higher performance rates for this class and the allover evaluation once having access to more training data for Phoma.

Figure 1. Cell phone camera images of sugar beet leaves showing leaf spots caused by Cercospora beticola (cerc), Ramularia beticola (ram), Pseudomonas syringae (pseu), Uromyces betae (rust), Phoma betae (phom), and combined infestation of Cercospora and Phoma (from upper left to lower right image)

The confusion matrix for binary classification is defined as follows:

prediction \ true class	+1	-1
+1	TP	FP
-1	FN	TN

TP is the number of true positive examples, TN is the number of true negative examples, FP is the number of false positive examples and FN is the number of false negative examples.

Define $P = TP + FN$ (all positive examples in D) and $N = TN + FP$ (all negative examples in D). Then,

- false positive rate: $FPR = \frac{FP}{TN+FP} = \frac{FP}{N}$
- false negative rate: $FNR = 1 - TPR = \frac{FN}{P}$
- *specificity* (true negative rate): $TNR = \frac{TN}{TN+FP} = \frac{TN}{N}$
- *sensitivity* (*recall*, true positive rate): $TPR = \frac{TP}{TP+FN} = \frac{TP}{P}$

- *precision* (positive predictive value): $PPV = \frac{TP}{TP+FP}$
- accuracy: $\frac{TP+TN}{P+N} = \frac{TP+TN}{n}$
- F-score (harmonic mean of precision and recall): $2\frac{1}{\frac{1}{prec}+\frac{1}{rec}} = 2\frac{precision*recall}{precision+recall}$

Exercise 2.1. Three different use cases.

(a) *Spam Filter*: What are false positive predictions in the spam filter use case? What are false negative predictions? Which event (FN or FP) is more costly for the user? Discuss!

(b) *Recommendation*: Define precision and recall for the specific use case of movie recommendation. What would it mean to maximize recall (precision) for this application? Should we maximize recall or precision? Discuss!

(c) *Information Retrieval*: Define precision and recall for the specific use case of internet search (search results presented by a earch engine). What would it mean to maximize recall (precision) for this application? Should we maximize recall or precision? Discuss!

2.2.3 Area under the ROC curve (AUC)

Oftentimes, our predictor has a threshold parameter t for when to decide whether the test point should be labeled positive or negative. E.g., predict $y^* = +1$ if $p(y^* = +1 \mid \mathbf{x}^*, X, \mathbf{y}) \geq t$. Usually t is set to 0.5, but that does not have to be the case, especially if there are differnt costs assigned to FN vs. FP predictions.

The ROC (*receiver operating characteristics*) curve plots the trade-off between true positive rate and false positive rate for all possible values of t. To summarize the curve in one measure we can compute the area under the ROC curve (AUC). In the example shown in Figure 1 using the squared loss in logistic regression has a slightly higher AUC value as using the other two losses.

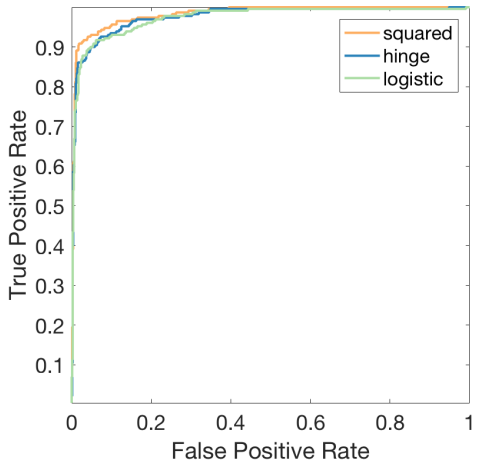


Figure 1: ROC for logistic regression using three different loss functions.

Interpretation: $p(y^* = +1 \mid \mathbf{x}^*, X, \mathbf{y})$ gives a ranking of our training data. AUC = probability that a randomly selected positive example is ranked higher than a randomly selected negative example.

Question: What value is the AUC for a perfect model?

Some properties of AUC:

- takes class imbalance in the test data into account

- only for binary classification (not for regression or multi-class)
- ROC curves may cross
- the thresholds mean different things for different models

2.2.4 Other performance measures for classification

- RMSE (for probabilistic classifier)
- information score
- cost curves

2.3 Summary of Performance Measures for Quality

- accuracy is not appropriate if
  - high class imbalance
  - FP and FN are not equally important
- confusion matrix provides all information (can be extended to **multi-class classification**)
- use RMSE for regression
- use negative log predictive density for a probabilistic regression model to incorporate predictive uncertainty in your evaluation/comparison

2.4 Efficiency

2.4.1 Runtime

Another aspect of performance for ML methods is **runtime**. Here we typically distinguish between *training time* and *test/prediction time*.

Example Performance Evaluation Scenarios

Compare runtime for different kernel computations for different applications with the goal to find the **best kernel** or show that your novel kernel is efficiently computable (*training time*).

Table 7 Attributed graphs: runtimes

Method	Dataset						
	SYNTHETIC	ENZYMES	PROTEINS	PRO_FULL	BZR	COX2	DHFR
PK	**0.3″**	**1.2′**	**20.0″**	**9.0′**	**3.0″**	**3.5″**	**15.1″**
P2K	17.2″	6.7′	31.4′	30.6′	1.3′	1.8′	5.9′
SP	13.7h	1.0h	6.8h	7.0h	20.5′	32.9′	1.6h
GH	16.9′	9.8′	1.2h	1.2h	7.8′	11.7′	31.2′
CSM	56.8h	21.8′	–	–	48.8′	1.6h	3.7h

Kernel computation times (cputime) are given in sec (x″), min (x′), or hours (xh). For all PKs, $t_{MAX} = 10$; for WL, $h_{MAX} = 10$; and for CSM, $k = 7$. All computations are performed on machines with 3.4 GHz Intel core i7 processors. Note that CSM is implemented in Java, so comparing computation times is only possible to a limited extent. OUT OF TIME means that the computation did not finish within 72h
Bold indicates the method fastest method

Compare runtime for different kernel computations on synthetic data to show scaling behavior with respect to growing data with the goal to find the **best kernel** or show that your novel kernel is efficiently computable (*training time*). Don't forget to report on theoretical runtime complexities as well:

- CWK: $\mathcal{O}(k\, t_{MAX}\, |E|\, n)$,
- LP: $\mathcal{O}(k\, N_{iter}\, |E|)$,
- DIFF: $\mathcal{O}(n^3)$, and
- L+: $\mathcal{O}(n^3)$,

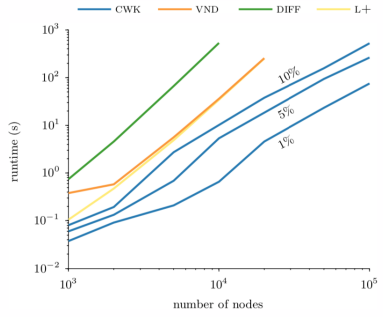


Figure 7.6: **Runtimes on PP-xk.** We show the loglog plot of number of nodes versus runtimes for the kernel computations of all kernels on graphs (CWK with 1%, 5% and 10% labeled nodes, VND, DIFF, and L+) on PP-xk where $x \in \{10^3, 2 \times 10^3, 5 \times 10^3, \ldots, 10^5\}$.

How about looking at both predictive quality and efficiency at the same time? Plot rubntime versus quality!

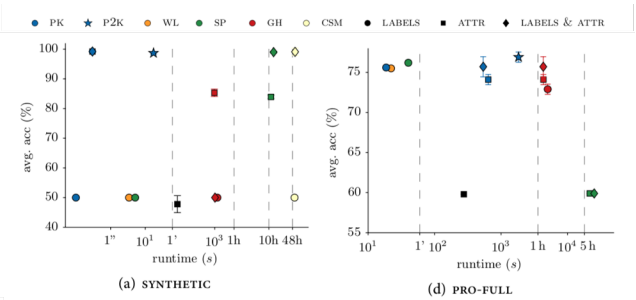


Figure 4.8: **Log Runtime vs. Accuracy on Attributed Graphs.** The plots show log(runtimes) in seconds plotted against average accuracy (± standard deviation). Methods are encoded by color and the used information (labels, attributes or both) is encoded by shape. Note that P2K also uses both, labels and attributes, however in contrast to PK both are propagated. For PRO-FULL the CSM kernel was OUT OF MEMORY meaning that 32 GB were exceeded. On SYNTHETIC CSM using the attribute information only could not be computed within 72h.

2.4.2 Memory

And last but not least we also want to look at **memory/storage** requirements.

Again look at different sizes of your (training) dataset to show how things scale.

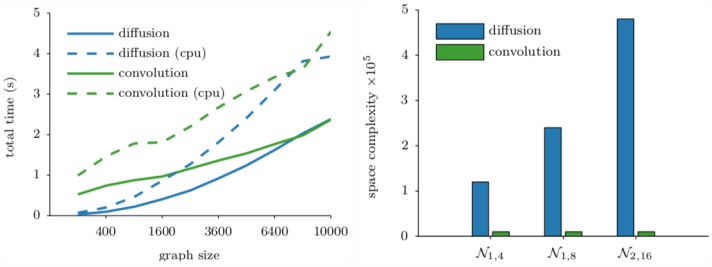


Figure 5.2: **Time and Space Complexity Analysis.** Comparison of two different implementations of information propagation (diffusion and convolution) in the propagation kernel computation for a synthetic dataset of 100 grid graphs of varying sizes. Panels (a-b) compare runtime complexities (cputime and actual elapsed time) of the kernel computation for different neighborhoods $\mathcal{N}_{r,p}$ and panel (d) illustrates the memory usage for one grid graph with 100 000 nodes.

3 Statistical Tests

Question: Can we attribute the performance to the classifier/model or is it due to chance?

Goal: figure out whether the evaluation results are representative for the general behavior of the classifier.
⇒ hypothesis testing to compare two models A and B.

Null hypothesis H_0: A and B perform equivalently.
Aim: reject this null hypothesis.

Approach:

1. choose appropriate test
2. compute test statistic
3. if in critical region, then reject H_0

Note: hypothesis testing is not a proof! It just gives some evidence. Do not overvalue the result of statistical tests. It is always possible to show that two models are significantly different (even if the difference is really small). We just have to run enough experiments or have enough data.

Choosing the *right* test is **as important as** choosing the *right* performance measure.

3.1 Parametric Tests

- makes stronger assumptions about the distribution of the underlying data
- might be tricky to verify that all assumptions hold

3.1.1 *t*-Test

The ***t*-test** is applied when we have two matched (*paired*) samples, i.e., the results for both models need to come from the same data with *matching* randomization and partitions.

CV \ classifier	A	B
$split_1$	acc_1^A	acc_1^B
$split_2$	acc_2^A	acc_2^B
$split_3$	acc_3^A	acc_3^B
$\vdots$	$\vdots$	$\vdots$
$split_n$	acc_n^A	acc_n^B

Table 1: Accuracy of model A and B in runs of cross validation with means μ_A and μ_B.

⇒ paired test as samples are not independent.
⇒ we test whether the two samples come from the same population.
⇒ we look at the difference in observed means and standard deviations.

H_0: the *observed* means μ_A and μ_B are the same (*both models have same performance*)

We assume that under this null hypothesis the *test statistic*

$$t = \frac{\bar{d}}{\hat{\sigma}_d/\sqrt{n}}, \tag{11}$$

follows a *t*-distribution, where

$$\bar{d} = \mu_A - \mu_B$$
$$\hat{\sigma}_d = \sqrt{\frac{\sum(d_i - \bar{d})^2}{n-1}}$$

with $d_i = acc_i^A - acc_i^B$ and degrees of freedom $df = n - 1$. The *two-tailed* version of the significance test can now be used to check if we can reject H_0. Use the table in Figure 3 to get the critical values $-t_\alpha$ and t_α for a (user-chosen) significance level α. If the test statistic t as computed via Eq. (11) falls into the critical region, i.e. $t \leq -t_\alpha$ and $t \geq t_\alpha$, cf. Fig. 2, we can reject H_0 at a significance level α. Typically $\alpha = 0.05$ or smaller.

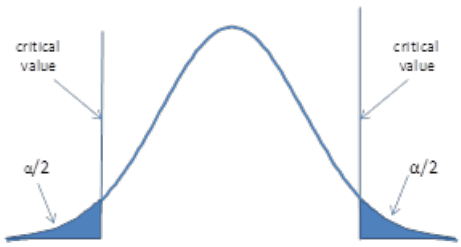


Figure 2: Critical region containing α of the probability mass for a *two-tailed* test.

The *one-tailed* version can be used to test whether the mean of the accuracies of model A is larger than that of model B (or vice versa).[1] Do **not** switch to the one-tailed version simply for the reason to get larger critical regions!

Assumptions:

(1) Samples are **continuous**.

(2) **Normality**: the samples d_i are (approximately) normally distributed → alternatively, the sample size (number of experiments/trials) should be greater than 30. Note that the paired sample t-test is robust to this being violated.

(3) **Randomness of the samples**: samples need to represent the population → achieve this by randomly selecting test sets or splitting the data using (randomly generated) cross-validation splits. **Caution**: the folds are overlapping ⇒ the samples are not random. Better use a multiple re-sampling CV scheme: 5×2-fold CV or 10×10-fold CV (or use McNemar's test).

3.2 Non-parametric Tests

- makes weaker assumptions
- less *powerful*[2] than parametric tests

For example, sign-test, McNemar's test and Wilcoxon signed-rank test.

3.2.1 McNemar's Test

The McNemar's test is the non-parametric counterpart of the t-test for classification tasks/experiments with categorical outcome (for continuous samples use the Wilcoxon signed-rank test or sign test).
Let c_{01} be the number of instances missclassified by A and correctly classified by B and c_{10} be the number of instances missclassified by B and correctly classified by A. McNemar's test tests the following null hypothesis:

H_0: $p(c_{01}) = p(c_{10})$ (*both models have same performance*)

The test statistic is given as:

$$\chi^2_{MC} = \frac{(|c_{01} - c_{10}| - 1)^2}{c_{01} + c_{10}}. \tag{12}$$

Requirement: $c_{01} + c_{10} \geq 20$
⇒ Under the null hypothesis χ^2_{MC} has a Chi-squared distribution with 1 degree of freedom. Find an illustration of the critical region and the table for $\chi^2_{df,\alpha}$ in Figure 4.

Exercise 3.1. Let $\alpha = 0.01$, what is $\chi^2_{1,\alpha}$?
For $\chi^2_{MC} = 7.1$, can we reject H_0? What does that mean in terms of which method (A or B) is better and how confident we should be about this?

3.2.2 Sign Test

The Sign test is usually used on multiple domains (though can be used on a single one with several trial – e.g., 10-fold cross-validation or re-sampling).
Define the following:
A_{win} = number of experiments A outperforms B

[1] See this discussion of one vs. two tailed tests: https://stats.idre.ucla.edu/other/mult-pkg/faq/general/faq-what-are-the-differences-between-one-tailed-and-two-tailed-tests/.
[2] The power of a statistical significance test is defined as the probability that it will reject a false null hypothesis.

B_{win} = number of experiments B outperforms A

H_0: $p(A_{win} > B_{win}) = 0.5$ (*both models have same performance*)

If H_0 is true, then number of wins follows a binomial distribution $B(n, \theta = 0.5)$. Use *two-tailed* version of the test and get the critical values w_α from the table in Figure 5. Use *one-tailed* if you want to test whether the number of wins of A is larger than that of model B (ore vice versa).
A_{win} needs to be larger than w_α to be considered statistically significantly better at a significance level of α. If $m > 25$, we can use the normal approximation of the binomial distribution.

Exercise 3.2. Consider the following results of an experiment comparing multiple ML methods on multiple datasets:

Dataset	NB	SVM	Adaboost	Rand Forest
Anneal	96.43	99.44	83.63	99.55
Audiology	73.42	81.34	46.46	79.15
Balance Scale	72.30	91.51	72.31	80.97
Breast Cancer	71.70	66.16	70.28	69.99
Contact Lenses	71.67	71.67	71.67	71.67
Pima Diabetes	74.36	77.08	74.35	74.88
Glass	70.63	62.21	44.91	79.87
Hepatitis	83.21	80.63	82.54	84.58
Hypothyroid	98.22	93.58	93.21	99.39
Tic-Tac-Toe	69.62	99.90	72.54	93.94

(a) Test NB vs SVM using the sign test. Can you reject the null-hypothesis stating that NB and SVM perform similarly on these data sets for $\alpha = 0.05$?

(b) Test NB vs SVM using the sign test. Can you reject the null-hypothesis stating that SVM performs better than NB for $\alpha = 0.05$?

(c) Test ADABOOST vs RANDFOREST using the sign test. Can you reject the null-hypothesis for $\alpha = 0.05$?

(d) Test ADABOOST vs RANDFOREST using the sign test. Can you reject the null-hypothesis stating that RF performs better than Ada for $\alpha = 0.05$?

3.2.3 [optional] Other Alternatives

Another test you might want to use is Wilcoxon's signed-Rank test. Wilcoxon's signed-Rank Test, like the sign test, deals with two classifiers on multiple domains or continuous samples from CV or re-sampling. It is also non-parametric, however, it is more powerful than the sign test.

For the case of multiple classifiers and multiple domains, two alternatives are possible. The parametric alternative is (one-way repeated measure) ANOVA and the non-parametric alternative is Friedman's Test.

3.2.4 Multiple Re-sampling

In practice, we often use **10 runs** of 10-fold cross-validation (10×10-fold CV), where each 10-fold CV run is considered as one experiment and we compare the average results of these runs in a statistical test. This procedure is called *multiple re-sampling* and gives **more stable estimates** of a method's performance than

using the results of the cross validation splits directly (those have a higher variance!). However, multiple re-sampling is time consuming. Another commonly used scheme is a 5×2-fold CV, which is more efficient.

4 Presenting Experimental Results

Throughout the notes we gave you a lot of examples on how to present experimental results. We find that presenting experimental results in a *meaningful*, *convincing*, and *fair* way is very challenging. Incorporating *summary statistics* and *plots* is extremely helpful for the reader (reviewers of a scientific publication, your boss, customers, etc.). Always take into account to whom you are presenting the result and always explain all measures and plots very carefully.

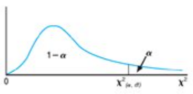
Appendix: Tables

Table T Critical Values of the *t* Distribution

df	One-Tail = .4 Two-Tail = .8	.25 .5	.1 .2	.05 .1	.025 .05	.01 .02	.005 .01	.0025 .005	.001 .002	.0005 .001
1	0.325	1.000	3.078	6.314	12.706	31.821	63.657	127.32	318.31	636.62
2	0.289	0.816	1.886	2.920	4.303	6.965	9.925	14.089	22.327	31.598
3	0.277	0.765	1.638	2.353	3.182	4.541	5.841	7.453	10.214	12.924
4	0.271	0.741	1.533	2.132	2.776	3.747	4.604	5.598	7.173	8.610
5	0.267	0.727	1.476	2.015	2.571	3.365	4.032	4.773	5.893	6.869
6	0.265	0.718	1.440	1.943	2.447	3.143	3.707	4.317	5.208	5.959
7	0.263	0.711	1.415	1.895	2.365	2.998	3.499	4.029	4.785	5.408
8	0.262	0.706	1.397	1.860	2.306	2.896	3.355	3.833	4.501	5.041
9	0.261	0.703	1.383	1.833	2.262	2.821	3.250	3.690	4.297	4.781
10	0.260	0.700	1.372	1.812	2.228	2.764	3.169	3.581	4.144	4.587
11	0.260	0.697	1.363	1.796	2.201	2.718	3.106	3.497	4.025	4.437
12	0.259	0.695	1.356	1.782	2.179	2.681	3.055	3.428	3.930	4.318
13	0.259	0.694	1.350	1.771	2.160	2.650	3.012	3.372	3.852	4.221
14	0.258	0.692	1.345	1.761	2.145	2.624	2.977	3.326	3.787	4.140
15	0.258	0.691	1.341	1.753	2.131	2.602	2.947	3.286	3.733	4.073
16	0.258	0.690	1.337	1.746	2.120	2.583	2.921	3.252	3.686	4.015
17	0.257	0.689	1.333	1.740	2.110	2.567	2.898	3.222	3.646	3.965
18	0.257	0.688	1.330	1.734	2.101	2.552	2.878	3.197	3.610	3.922
19	0.257	0.688	1.328	1.729	2.093	2.539	2.861	3.174	3.579	3.883
20	0.257	0.687	1.325	1.725	2.086	2.528	2.845	3.153	3.552	3.850
21	0.257	0.686	1.323	1.721	2.080	2.518	2.831	3.135	3.527	3.819
22	0.256	0.686	1.321	1.717	2.074	2.508	2.819	3.119	3.505	3.792
23	0.256	0.685	1.319	1.714	2.069	2.500	2.807	3.104	3.485	3.767
24	0.256	0.685	1.318	1.711	2.064	2.492	2.797	3.091	3.467	3.745
25	0.256	0.684	1.316	1.708	2.060	2.485	2.787	3.078	3.450	3.725
26	0.256	0.684	1.315	1.706	2.056	2.479	2.779	3.067	3.435	3.707
27	0.256	0.684	1.314	1.703	2.052	2.473	2.771	3.057	3.421	3.690
28	0.256	0.683	1.313	1.701	2.048	2.467	2.763	3.047	3.408	3.674
29	0.256	0.683	1.311	1.699	2.045	2.462	2.756	3.038	3.396	3.659
30	0.256	0.683	1.310	1.697	2.042	2.457	2.750	3.030	3.385	3.646
40	0.255	0.681	1.303	1.684	2.021	2.423	2.704	2.971	3.307	3.551
60	0.254	0.679	1.296	1.671	2.000	2.390	2.660	2.915	3.232	3.460
120	0.254	0.677	1.289	1.658	1.980	2.358	2.617	2.860	3.160	3.373
∞	0.253	0.674	1.282	1.645	1.960	2.326	2.576	2.807	3.090	3.291

Source: From *Biometrika Tables for Statisticians*, Vol. 1, Third Edition, edited by E. S. Pearson and H. O. Hartley, 1966, p. 146. Reprinted by permission of the Biometrika Trustees.

Figure 3: t-distribution used in t-test.



Degrees of Freedom	Upper Tail Areas (α)											
	.995	.99	.975	.95	.90	.75	.25	.10	.05	.025	.01	.005
1			0.001	0.004	0.016	0.102	1.323	2.706	3.841	5.024	6.635	7.879
2	0.010	0.020	0.051	0.103	0.211	0.575	2.773	4.605	5.991	7.378	9.210	10.597
3	0.072	0.115	0.216	0.352	0.584	1.213	4.108	6.251	7.815	9.348	11.345	12.838
4	0.207	0.297	0.484	0.711	1.064	1.923	5.385	7.779	9.488	11.143	13.277	14.860
5	0.412	0.554	0.831	1.145	1.610	2.675	6.626	9.236	11.071	12.833	15.086	16.750
6	0.676	0.872	1.237	1.635	2.204	3.455	7.841	10.645	12.592	14.449	16.812	18.548
7	0.989	1.239	1.690	2.167	2.833	4.255	9.037	12.017	14.067	16.013	18.475	20.278
8	1.344	1.646	2.180	2.733	3.490	5.071	10.219	13.362	15.507	17.535	20.090	21.955
9	1.735	2.088	2.700	3.325	4.168	5.899	11.389	14.684	16.919	19.023	21.666	23.589
10	2.156	2.558	3.247	3.940	4.865	6.737	12.549	15.987	18.307	20.483	23.209	25.188
11	2.603	3.053	3.816	4.575	5.578	7.584	13.701	17.275	19.675	21.920	24.725	26.757
12	3.074	3.571	4.404	5.226	6.304	8.438	14.845	18.549	21.026	23.337	26.217	28.299
13	3.565	4.107	5.009	5.892	7.042	9.299	15.984	19.812	22.362	24.736	27.688	29.819
14	4.075	4.660	5.629	6.571	7.790	10.165	17.117	21.064	23.685	26.119	29.141	31.319
15	4.601	5.229	6.262	7.261	8.547	11.037	18.245	22.307	24.996	27.488	30.578	32.801
16	5.142	5.812	6.908	7.962	9.312	11.912	19.369	23.542	26.296	28.845	32.000	34.267
17	5.697	6.408	7.564	8.672	10.085	12.792	20.489	24.769	27.587	30.191	33.409	35.718
18	6.265	7.015	8.231	9.390	10.865	13.675	21.605	25.989	28.869	31.526	34.805	37.156
19	6.844	7.633	8.907	10.117	11.651	14.562	22.718	27.204	30.144	32.852	36.191	38.582
20	7.434	8.260	9.591	10.851	12.443	15.452	23.828	28.412	31.410	34.170	37.566	39.997
21	8.034	8.897	10.283	11.591	13.240	16.344	24.935	29.615	32.671	35.479	38.932	41.401
22	8.643	9.542	10.982	12.338	14.042	17.240	26.039	30.813	33.924	36.781	40.289	42.796
23	9.260	10.196	11.689	13.091	14.848	18.137	27.141	32.007	35.172	38.076	41.638	44.181
24	9.886	10.856	12.401	13.848	15.659	19.037	28.241	33.196	36.415	39.364	42.980	45.559
25	10.520	11.524	13.120	14.611	16.473	19.939	29.339	34.382	37.652	40.646	44.314	46.928
26	11.160	12.198	13.844	15.379	17.292	20.843	30.435	35.563	38.885	41.923	45.642	48.290
27	11.808	12.879	14.573	16.151	18.114	21.749	31.528	36.741	40.113	43.194	46.963	49.645
28	12.461	13.565	15.308	16.928	18.939	22.657	32.620	37.916	41.337	44.461	48.278	50.993
29	13.121	14.257	16.047	17.708	19.768	23.567	33.711	39.087	42.557	45.722	49.588	52.336
30	13.787	14.954	16.791	18.493	20.599	24.478	34.800	40.256	43.773	46.979	50.892	53.672

Figure 4: Chi-squared distribution used in McNemar's test.

Level of significance α					Level of significance α				
Two-sided	0.10	0.05	0.02	0.01	Two-sided	0.10	0.05	0.02	0.01
One-sided	0.05	0.025	0.01	0.005	One-sided	0.05	0.025	0.01	0.005
n					n				
1	–	–	–	–	31	11	13	15	17
2	–	–	–	–	32	12	14	16	16
3	–	–	–	–	33	11	13	15	17
4	–	–	–	–	34	12	14	16	16
5	5	–	–	–	35	11	13	15	17
6	6	6	–	–	36	12	14	16	18
7	7	7	7	–	37	11	13	17	17
8	6	8	8	8	38	12	14	16	18
9	7	7	9	9	39	13	15	17	17
10	8	8	10	10	40	12	14	16	18
11	7	9	9	11	45	13	15	17	19
12	8	8	10	10	46	14	16	18	20
13	7	9	11	11	49	13	15	19	19
14	8	10	10	12	50	14	16	18	20
15	9	9	11	11	55	15	17	19	21
16	8	10	12	12	56	14	16	18	20
17	9	9	11	13	59	15	17	19	21
18	8	10	12	12	60	14	18	20	22
19	9	11	11	13	65	15	17	21	23
20	10	10	12	14	66	16	18	20	22
21	9	11	13	13	69	15	19	23	25
22	10	12	12	14	70	16	18	22	24
23	9	11	13	15	75	17	19	23	25
24	10	12	14	14	76	16	20	22	24
25	11	11	13	15	79	17	19	23	25
26	10	12	14	14	80	16	20	22	24
27	11	13	13	15	89	17	21	23	27
28	10	12	14	16	90	18	20	24	26
29	11	13	15	15	99	19	21	25	27
30	10	12	14	16	100	18	22	26	28

Source: Wijvekate, 1962

Figure 5: Binomial distribution used in sign-test.