ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

Μηχανική μάθηση

Ενότητα 8: Model Selection and Performance Estimation

Ιωάννης Τσαμαρδίνος
Τμήμα Επιστήμης Υπολογιστών

Performance Estimation

- Need to produce a single, final model
- But also estimate its performance

- Why estimate performance
  - Know what to expect out of a model / system
  - Select the best model out of all possible models one could construct
  - Compare different learning algorithms

- Probably the most underestimated problem in machine learning, data mining, pattern recognition

Ideal Performance Estimation

1. Learn a model from samples in *S* (train-set)
2. Install the model in its intended operational environment
3. Observe its operation for some time, for new cases S'
4. Label with a gold-standard the cases in S' (test-set)
5. Estimate the performance of the model on S'

Ideal Performance Estimation

Golden Rule:

Simulate: learn from S, make operational, test on new samples S'

- Pros and cons?

Estimating the Error in the Training Set

- Why not?

Simulating the Ideal

- Randomly split original data
- Learn on Train
- Test on Test
- Called hold-out estimation
- Can it go wrong?

Train-Test Error and Complexity

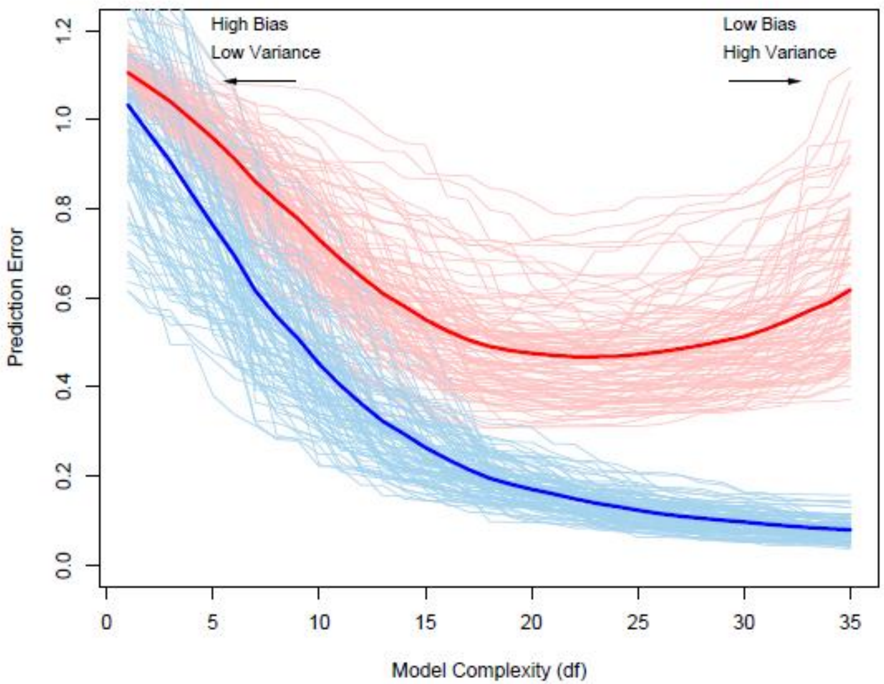


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for* 100 *training sets of size* 50 *each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$.

Overfitting and Underfitting

- No accepted definition
- Overfitting of a method or model: learning data characteristics (patterns) that do not generalize
  - More frequent for methods with small bias (can learn anything, make few assumptions about the data)
- Underfitting: not learning characteristics that would generalize
  - More frequent for methods with large bias (make strong assumptions about the data)
- Use of the term overfitting "the results are overfitted" means:
  - The performance **estimates** provided are optimistic due to overfitting of the data and poor estimation methods
  - The performance estimates may be optimistic due to methodological errors in their production

Notation

- f(**T**)
  - Model learnt on dataset **T** by a given learning method with specific parameter settings
  - E.g., decision tree with parameter MaxPChance = 0.05
- f(**x**, **T**)
  - Apply function f(**T**) on example **x** and obtain a prediction
  - Simplify as f(**x**) for given **T**
- Loss function: measures the discrepancy between truth and prediction
  - L(y, f(**x**))

Loss Functions

- Regression (y is continuous)
  - $L(y, f(x)) = (y - f(x))^2$ (squared error)
  - $L(y, f(x)) = |y - f(x)|^2$ (absolute error)
- Categorization (y is discrete)
  - $L(y, f(x) = I(y \neq f(x))$ (zero-one loss, 1 – accuracy)
- Conditional Density Estimation (y is discrete, prediction is the conditional probability for each possible value of y)
  - $L(y, f(x)) = -2 \log f_y(x)$ (probability given to the true class)
- AUC not easily expressed as a loss function (depends on the whole dataset, not a single example)

Sample Mean Loss

- Define

$$L(f, Test) = \frac{1}{|Test|} \sum_{<y,\mathbf{x}> \in Test} L(y, f(x))$$

- L(f, Test) could be defined as the AUC on the Test

Hold-Out Estimation

- Hold-Out(Data **D**)
  - Randomly split **D** to Train and Test
  - Returned Model
    - f(Train)
  - Performance Estimation
    - L(f(Train), Test)

- Pros: simple, computationally efficient, and correct
- Pros: appropriate when data are plenty
- Cons: some data are not “lost” to estimation

Presumed Learning Curve

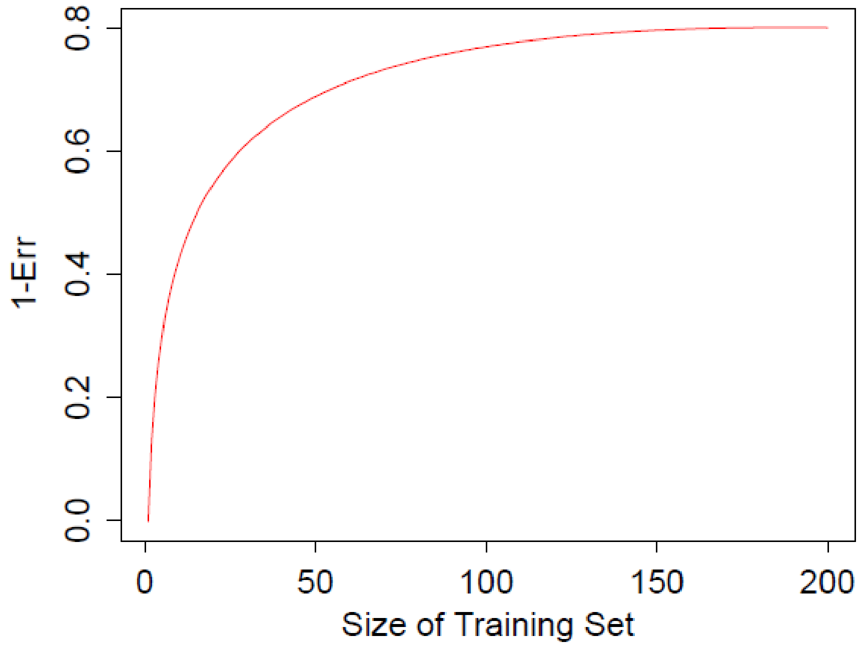


FIGURE 7.8. *Hypothetical learning curve for a classifier on a given task: a plot of* $1 - \text{Err}$ *versus the size of the training set* N*. With a dataset of* 200 *observations, 5-fold cross-validation would use training sets of size* 160*, which would behave much like the full set. However, with a dataset of* 50 *observations fivefold cross-validation would use training sets of size* 40*, and this would result in a considerable overestimate of prediction error.*

Hold-Out Estimation Revisited

- Why not train on all data? The new model should be better

- Hold-Out2(Data **D**)
  - Randomly split **D** to Train and Test
  - Returned Model
    - f(**D**)
  - Performance Estimation
    - L(f(Train), Test)

- Our estimation is NOT directly computed on the returned model
- Then, what do we estimate?

Performance? What Performance?

- For a given **model** f, true Loss of model f
  - $L(f) = E_{<y,\mathbf{x}>} (L(y, f(\mathbf{x}))$
  - Expectation taken over all possible samples

- For a given learning method f, true loss of the method f when trained on datasets of size N
  - $L_N(f) = E_{T, \; |T| = N} \; L(f(T))$
  - Expectation taken over all possible training datasets of size N

Hold-Out Estimation Again

- Hold-Out
  - Model: f(Train)
  - Estimate: L(f(Train), Test) (sample mean loss on the test set)
  - Estimates: L(f(Train)) (the true loss of the model)

- Hold-Out2
  - Model: f(All)
  - Estimate: L(f(Train), Test) (sample mean loss on the test set)
  - Estimates: L_N (f) (≈ L(f(Train), Test))
    - True loss of the learning **method** when trained on datasets of size N = |Train|
    - Conservative estimation of L_M (f) , M = |All|

Train-Test Splitting Decisions

- Large Train – Small Test
  - L(f(Train), Test) ≈ L_N (f) closer to the ideal L_M (f)
    - Estimate less conservative
  - L(f(Train), Test) has larger variance
    - Estimate less reliable
- Small Train –Large Test
  - Estimate more conservative
  - Estimate more reliable
- Typical splits: Train set is 66%, 75%, 80% of the data

K-Fold Cross-Validation

- Idea:
  - L(f(Train), Test) is a single sample to estimate L_N(f(Train))
  - Can we have more samples for a better estimation?
  - We would need more **independent** Test sets
  - Any ideas?

K-Fold Cross-Validation

Train	Train	Train	Test	Train

- Split to K-folds
- F(j) the samples of the j-th fold
- Cross-Validation(Data **D**, number K)
  - Randomly split **D** to K folds
  - Returned Model
    - ??
  - Performance Estimation

$$CVL(D,K) = \frac{1}{K}\sum_{j=1}^{K} L(f(D \setminus F(j)), F(j))$$

K-Fold Cross-Validation

- Split to K-folds
- F(j) the samples of the j-th fold
- Cross-Validation(Data **D**, number K)
  - Randomly split **D** to K folds
  - Returned Model
    - f(**D**)
  - Performance Estimation

$$CVL(D,K) = \frac{1}{K}\sum_{j=1}^{K} L(f(D \setminus F(j)), F(j))$$

What Performance?

- What quantity does CV estimates?

What Performance?

- What quantity does CV estimates?
  - L(f(Train), Test) is a single sample to estimate L_N (f)
  - CVL(D, K) estimates L_N (f) with more samples (test sets)

- Connection between returned model and estimate
  - f(**D**): model produced by method f on a datasets of size M = |**D**|
  - CVL(D, K): estimate of L_N (f), the expected loss of the method f when trained on datasets of size N = M - M / K
  - CVL is conservative since N < M

How K Affects Estimation

$$CVL(D,K) = \frac{1}{K}\sum_{j=1}^{K} L(f(D \setminus F(j)), F(j))$$

- As K increases CVL(D, K)
  - Sums over more test-sets so it becomes more reliable (smaller variance)
  - (at the same time) Sums over less reliable estimates (each test set is smaller), so it becomes less reliable (larger variable)
  - The less conservative it becomes; **depends where we are on the learning curve of the classifier**
  - The learnt models f(D\F(j)) become more correlated and we tend to estimate the expected loss **given** the specific dataset
  - The more computational resources it requires

Typical Ks

- Typical values for K
  - 3, 5, 10
  - N (**Leave One Out**)

- Evidence that Leave-One-Out is not always the best choice

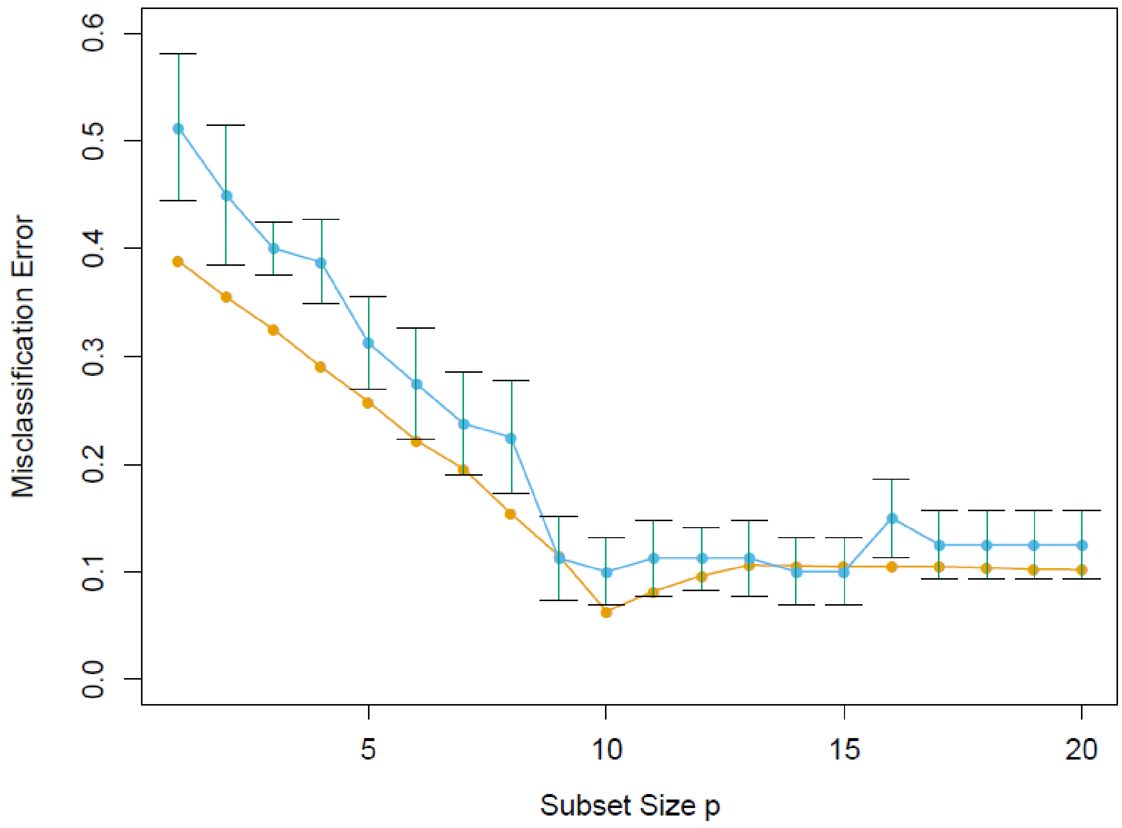


FIGURE 7.9. *Prediction error (orange) and tenfold cross-validation curve (blue) estimated from a single training set, from the scenario in the bottom right panel of Figure 7.3.*

Pitfalls of Cross-Validation

Golden Rule:

Simulate: learn from S, make operational, test on new samples S’

- Scale data so that each variable has zero mean and standard deviation of 1
- Remove variables independent of the target
- <model, est> = Cross-Validation(**D**, K)
- Tell the client that *model* is expected to have loss *est*

Pitfalls of Cross-Validatio

It peeks in the test cases!!!

Golden Rule:

Simulate: learn from S, make operational, test on new samples S'

- Scale data so that each variable has zero mean and standard deviation of 1
- Remove variables independent of the target
- <model, est> = Cross-Validation(**D**, K)
- Tell the client that *model* is expected to have loss *est*

Pitfalls of Cross-Validatio

It peeks in the test cases!!!

Golden Rule:

Simulate: learn from S, make o ial, test on new samples S'

- Scale data so that each variable has zero mean and standard deviation of 1
- Remove variables independent of the target
- <model, est
- Tell the clie est

Scaling and variable selection is part of the learning method; they **also** have to be CVed

Correct CV

Learning Method producing a model (function)

- Cross-Validation(Data **D**, number K)
  - Randomly split **D** to K folds
  - Returned Model: f(**D**)
  - Performance Estimation: $CVL(D,K) = \frac{1}{K}\sum_{j=1}^{K} L(f(D \setminus F(j)), F(j))$
- f(Data Train)
  - Normalize Train, store normalizing parameters normpar
  - Identify the most important variable-set S from Train
  - Project Train on S only
  - Learn a decision tree TR from Train data
  - Return a function f(x)
    - Normalizes x according to normpar
    - Retain only variables S from vector x
    - Return the output of TR on (modified vector) x

Learnt Model

Example of Overfitting due to Bad CV

- Consider a scenario with $N = 50$ samples in two equal-sized classes, and $p = 5000$ quantitative predictors (standard Gaussian) that are independent of the class labels. The true (test) error rate of any classifier is 50%. We carried out the above recipe, choosing in step (1) the 100 predictors having highest correlation with the class labels, and then using a 1-nearest neighbor classifier, based on just these 100 predictors, in step (2). Over 50 simulations from this setting, the average CV error rate was 3%. This is far lower than the true error rate of 50%.

- Hastie, Tibshirani, Friedman, Elements of Statistical Learning, p. 245, second edition

Stratified Cross-Validation

- Setting
  - Sample size 100
  - Equal frequency of 10 classes
  - 3-Fold Cross-Validation
- Problem:
  - Quite probable that some of the classes do not appear at all in a training set
- Solution: ?

Stratified Cross-Validation

- Setting
  - Sample size 100
  - Equal frequency of 10 classes
  - 3-Fold Cross-Validation

- Problem:
  - Quite probable that some of the classes do not appear at all in a training set
- Solution: Stratified Cross-Validation
  - Randomly split to fold, while maintaining the distribution of the classes as close as possible to the one in the full dataset
  - Highly recommended when lots of classes compared to sizes of the training sets
  - How to implement?

Model Selection and Parameter Optimization

- Several available classifiers with several possible parameters
  - K-NN
    - Parameter K, distance function
  - Simple Bayesian Classifier
    - Parameter l
  - Decision Trees
    - Parameter MaxPChance
- Several preprocessing algorithms
  - Feature selection
  - Imputation
  - Discretization
  - Normalization
- Different Representations!

- Which method + parameters to choose? Which final model?

Construct all Models, Select Best

- For each possible learning method f_i (combination of learner + parameters)
  - $<Perf_i , model_i> = \text{Hold-Out2}(D)$ (of method f_i)
- End for
- $j = \text{argmax } Perf_i$
- Return $<Perf_j , model_j>$

Construct all Models, Select Best

Algorithm	Parameter	Performance (Loss)
K-NN	K=1	0.81
	K=2	0.84
	K=5	0.88
DT	MaxPChance=0.01	0.83
	MaxPChance=0.05	0.9
	MaxPChance=0.1	0.81
SB	I = 0	0.75
	I=1	0.83

Construct all Models, Select Best

Train	Test

Algorithm	Parameter	Performance (Loss on Test)
K-NN	K=1	0.81
	K=2	0.84
	K=5	0.88
DT	MaxPChance=0.01	0.83
	MaxPChance=0.05	0.9
	MaxPChance=0.1	0.81
SB	I = 0	0.75
	I=1	0.83

Selected model

Construct all Models, Select Best

- For each possible learning method f_i (combination of learner + parameters)
  - $<Perf_i , model_i> = \text{Hold-Out}(D)$ (of method f_i)
- End for
- $j = \text{argmax } Perf_i$
- Return $<Perf_j , model_j>$

It peeks in the test cases to select the final model: violation of Golden Rule

Thought Experiment

- Hire investment advisor
- Test Task: predict tomorrow's stock market (up or down)
- Several possible candidates
- Test Set: next 14 days
- Hire if more than 11 out of 14 successes
- Assume 50-50% of marker going up or down each day

Hiring an Advisor

- 1 candidate
- Chances of hiring a charlatan
- S_i = successes of candidate i
- $P(S_1 > 10)$?

Hiring an Advisor

- 1 candidate
- Chances of hiring a charlatan
- S_i = successes of candidate i
- $P(S_1 \geq 11) = ?$
  - $P(S_1 = 11) = \binom{14}{11} \theta^{11} \theta^{14-11}$ (binomial distribution)
  - $P(S_1 \geq 11) = \sum_{i=11}^{14} \binom{14}{11} \theta^{k} \theta^{n-k} = 0.0287$
- Low chance of hiring a charlatan

Hiring an Advisor

- 10 candidates
- Chances of hiring a charlatan
- S_i = successes of candidate i
- $P(S_i \geq 11) = 0.0287$
- P(al least one $S_i \geq 11$) = ?

Hiring an Advisor

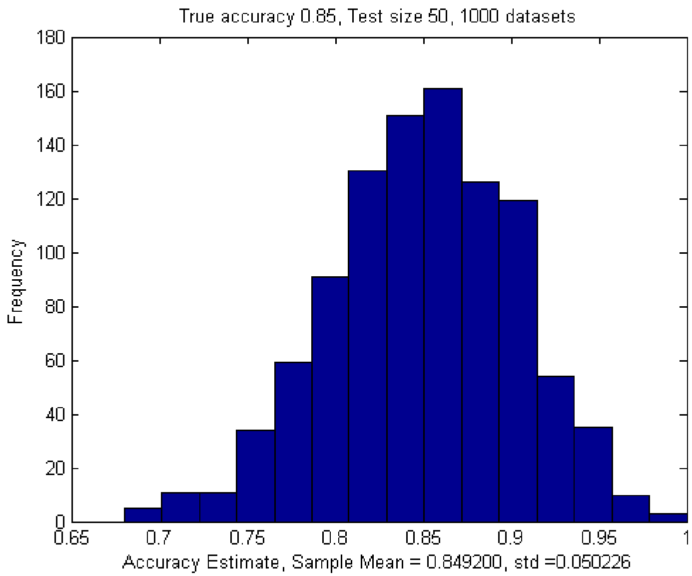
- 10 candidates
- Chances of hiring a charlatan
- S_i = successes of candidate i
- $\theta = P(S_i \geq 11) = 0.0287$
- P(al least one $S_i \geq 11) = 1 - (1- \theta)^{10} = 0.2525$

- High chances of hiring a charlatan.

- Why? What went wrong?

Extreme Distributions: 1 Model

Alg	Parameter	Loss
K-NN	K=1	



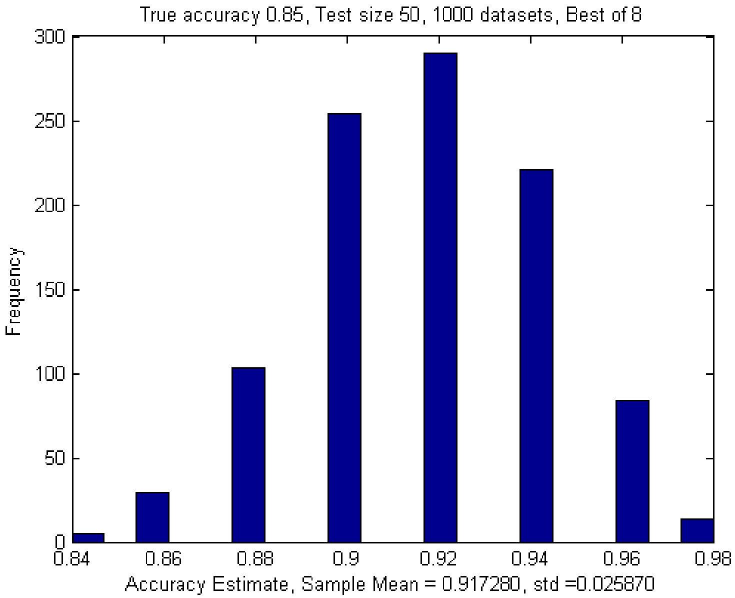
Assume: Equal true accuracies 85%

Mean = 0.85

Std = $\mathrm{sqrt}(N \cdot p \cdot (1-p)) / N = 0.0505$

Extreme Distributions: 8 models

Alg	Parameter	Loss
K-NN	K=1	
	K=2	
	K=5	
DT	MaxPChance=0.01	
	MaxPChance=0.05	
	MaxPChance=0.1	
SB	I = 0	
	I=1	



Assume: Equal true accuracies 85%
Mean, Std follow an Extreme Distribution

Choose Model AND Estimate Performance

- How?

Choose Model AND Estimate Performance

- Train: used to train model
- Validate: used to choose best model
- Test: used to estimate performance

Learning Procedure

- The Learning procedures **includes** selection of best parameter set

Parameterizing Model Construction

- Example with three possible algorithms

- BuildModel(Data Train, Parameter set **a**)
  - Switch a
    - Case $a_1 = 1$, *method* = K-NN, $K = a_2$
    - Case $a_1 = 2$, *method* = Decision Tree, $MaxPChance = a_2$
    - Case $a_1 = 3$, *method* = Simple Bayes, $I = a_2$
  - Return model (function f(x)) learnt by *method* using parameter *a* on Train data

Learning Method with Model Selection

- Learner-Validate(Data **D**, parameter sets **a**)
  - Partition **D** to Train, Validate
  - For each parameter set a_i
    - f_i = BuildModel(Train, a_i)
    - L_i = L(f_i, Validate)
  - End
  - Select best parameters: a^* = a's minimizing L_i
  - Return model BuildModel(**D**, a^*)

Model Selection and Estimation with Hold Outs

- Hold-Out(Data **D**, parameter sets **a**)
  - Randomly split **D** to Train+Validation and Test
  - Returned Model
    - Learner-Validate**(D**, **a**)
  - Performance Estimation
    - L(Learner-Validate(Train+Validation, **a**), Test)

Model Selection with CV

- Learner-CV(Data in folds F(j), parameter sets **a**)
  - For each parameter set a_i
    - $L_i = 0$
    - For each fold F(j)
      - Train = all folds but F(j), Validate = F(j)
      - f_i = BuildModel(Train, a_i)
      - $L_i = L_i + L(f_i$, Validate)
    - End
    - $L_i = L_i$ / #folds
  - End
  - Select best parameters: $a^* = $ a's minimizing L_i
  - Return model BuildModel(**D**, a^*)

Estimation with CV

- Nested-CV(Data **D**, parameter sets **a**)
  - Randomly split **D** to folds F(j)
  - Returned Model
    - Learner-CV(folds F(j), **a**)
  - Performance Estimation
    - L = 0
    - For each fold F(j)
      - f_i = Learner-CV(all folds but F(j), **a**)
      - L = L + L(f_i, F(j))
    - End
    - L = L / #folds
    - Return L

Experimentation Protocols

- Hold-out MS, Hold-out PE
- Cross-validated MS, Cross-Validated PE (nested cross validation)
- All 4 combinations are possible
- How many models does each one build?

Nested Cross Validation

- Not yet standard practice (one day though!)
- Important when many different combinations of learning parameters are attempted

Alternatives to Cross Validation

- There exist alternative ways to estimate performance for **model selection** than CV

  - AIC – Akaike Information Criterion
  - BIC – Bayesian Information Criterion
  - VC-dimension (Vapnik – Chervonenkis dimension)

- Quite interesting but outside the scope of this class

Download and Play

- Gene Expression Model Selector
  - mensxmachina.org
- Tool to automatically estimate performance and select models with a few clicks

Summary

- Would like to:
  - Learn a model
  - Select the best model we can construct (best parameter combination)
  - Estimate its performance
- Need to follow the golden rule so we do not produce optimistic estimates (biased)

What to Know

- Why not measure error in the training set
- Detect violations from the Golden Rule
- Different protocols for performance estimation
  - Hold-Out
  - Cross-Validation correctly applied
- Different protocols for performance estimation + model selection
- What quantity each estimation method approximates
- Trade-offs of each method

Τέλος Ενότητας

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Κρήτης**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Σημειώματα

Σημείωμα αδειοδότησης (1)

Σημείωμα αδειοδότησης (2)

- Ως **Μη Εμπορική** ορίζεται η χρήση:
  - που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
  - που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
  - που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

- Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

.

Σημείωμα Αναφοράς

Copyright Πανεπιστήμιο Κρήτης, Ιωάννης Τσαμαρδίνος 2015. «Μηχανική Μάθηση. Model Selection and Performance Estimation». Έκδοση: 1.0. Ηράκλειο 2015. Διαθέσιμο από τη δικτυακή διεύθυνση: https://opencourses.uoc.gr/courses/course/view.php?id=362.

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.