Testing classifier accuracy (cse352)

Professor Anita Wasilewska

Lecture Notes on Learning

Overview

- Introduction
- Basic Concept on Training and Testing
- Resubstitution (N ; N)
- Holdout (2N/3 ; N/3)
- x-fold cross-validation (N-N/x ; N/x)
- Leave-one-out (N-1 ; 1)

Introduction

Predictive Accuracy Evaluation

The main methods of predictive accuracy evaluations are:

- Resubstitution (N ; N)
- Holdout (2N/3 ; N/3)
- x-fold cross-validation (N-N/x ; N/x)
- Leave-one-out (N-1 ; 1)

where N is the number of records (instances) in the dataset

Training and Testing

- REMEMBER: we must know the classification (class attribute values) of all instances (records) used in the test procedure.
- **Basic Concepts**

Success: instance (record) class is predicted correctly

Error: instance class is predicted incorrectly

Error rate: a percentage of errors made over the whole set of instances (records) used for testing

Predictive Accuracy: a percentage of well classified data in the testing data set.

Training and Testing

- Example:

Testing Rules (testing record #1) = record #1.class - Succ
Testing Rules (testing record #2) not= record #2.class - Error
Testing Rules (testing record #3) = record #3.class - Succ
Testing Rules (testing record #4) = instance #4.class - Succ
Testing Rules (testing record #5) not= record #5.class - Error

Error rate:

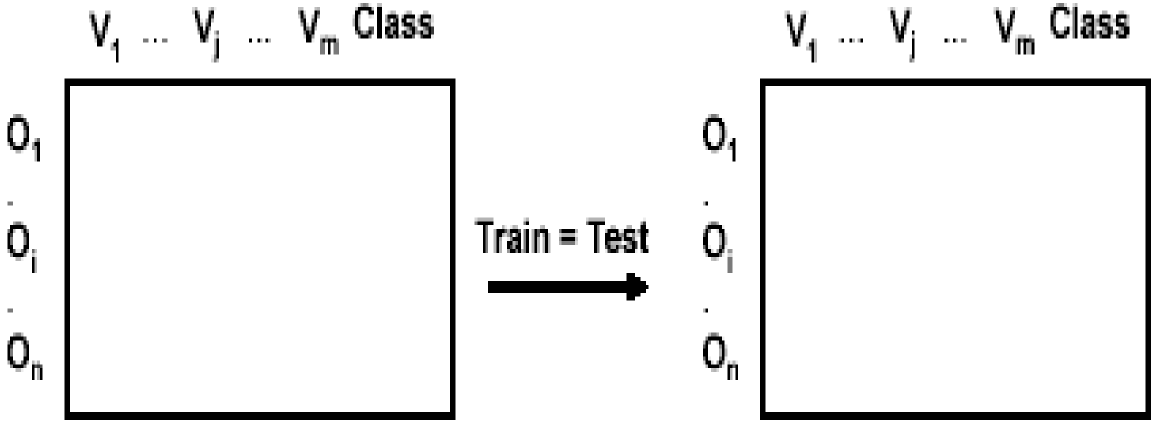
2 errors: #2 and #5

Error rate = 2/5=40%

Predictive Accuracy: 3/5 = 60%

Resubstitution (N ; N)

Testing the classification model by using the given data set (already used for „training")



Re-substitution Error Rate

- Re-substitution error rate is obtained from training data
- Training Data Error: uncertainty of the rules
- The error rate is not always 0%, but usually (and hopefully) very low!
- Resubstitution error rate indicates only how good (bad) are our results (rules, patterns, NN) on the TRAINING data; expresses some knowledge about the algorithm used.

Re-substitution Error Rate

- Re-substitution Error Rate is usually used as the performance measure:

The training error rate reflects imprecision of the training results: the lower, the better

Predictive accuracy reflects how good are the training results with respect to the test data: the higher, the better

(N:N) re-substitution does not compute predictive accuracy

- Re-substitution error rate = training data error rate

Why not always 0%?

- The error/error rate on the training data is not always 0% because algorithms involve different (often statistical) parameters and measures that lead to uncertainties
- It is used for “parameters tuning”
- The error on the training data is NOT a good indicator of performance on future data since it does not measure any not yet seen data.
- Solution:

Split data into training and test set

Training and test set

- Training and Test data may differ in nature, but must have the same format.

Example:

Given customer data from two different towns A and B.

We train the classifier with the data from town A and we test it on data from town B, and vice-versa

Training and test set

- It is important that the test data is not used in any way to create the training rules
- In fact, learning schemes operate in three stages:

  Stage 1: build the basic structure (training)

  Stage 2: optimize parameter settings; can use (N:N) re-substitution (parameter tuning)

  Stage 3: use test data to compute predictive accuracy/error rate

  Proper procedure uses three sets: *training data*, *validation data* and *test data*

- validation data is used for parameter tuning, not test data; validation data can be the training data, or a subset of training data.

  The test data cannot be used for parameter tuning!

Training and testing

- Generally, the larger is the training set, the better is the classifier
- The larger the test data the more accurate the predictive accuracy, or error estimation
- Remember: the error rate of Re-substitution(N;N) can tell us ONLY whether the algorithm used in the training is good or not, or how good it is.

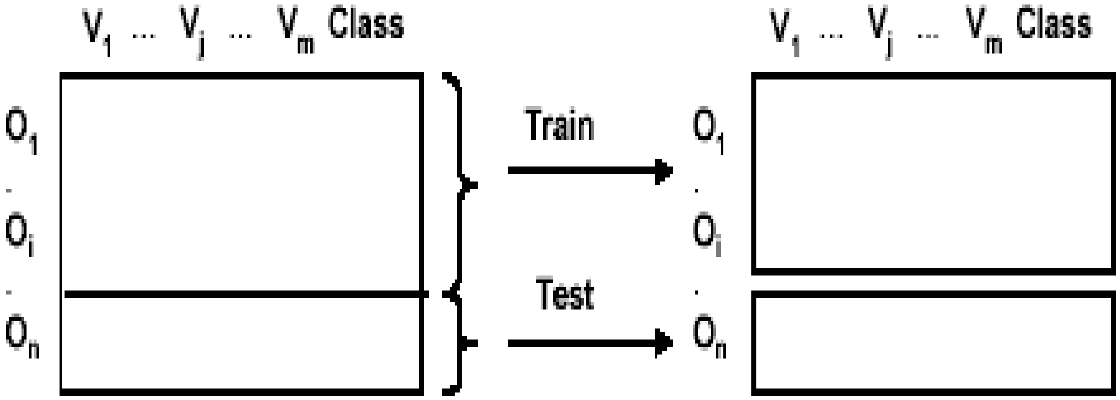
- ***Holdout* procedure:** a method of splitting original data into training and test set

- Dilemma: ideally both training and test set should be large! What to do if the amount of data is limited?
- How to split the data into training and test subsets (disjoint)?

Holdout

Train-and-Test (for large sample sizes) (> 1000))
dividing the given data set in

- a **training sample** for generating the classification model
- a **test sample** to test the model on independent objects with given classifications (randomly selected, 20-30% of the complete data set)



Holdout (2N/3 ; N/3)

- The holdout method reserves a certain amount of data for testing and uses the remainder for training – so they are disjoint!

- Usually, one third (N/3) of data is used for testing, and the rest (2N/3) for training

- The choice of records for the train and test
data is essential, so we usually perform a cycle:
Train-and-test; repeat

Repeated Holdout

- Holdout can be made more reliable by repeating the process with different sub-samples (subsets of data):

  1. In each iteration, a certain proportion is randomly selected for training, the rest of data is used for testing

  2. The error rates or predictive accuracy on the different iterations are averaged to yield an overall error rate, or predictive accuracy

- Repeated holdout still not optimum: the different test sets **overlap**

x-fold cross-validation (N-N/x ; N/x)

- The cross-validation is used to prevent the overlap of the test sets
- first step: split data into x disjoint subsets of equal size

second step: use each subset in turn for testing, the remainder for training

The error (predictive accuracy) estimates are averaged to yield an overall error (predictive accuracy) estimate

Cross-validation

- Standard cross-validation: 10-fold cross-validation
- Why 10?

Extensive experiments have shown that this is the best choice to get an accurate estimate. There is also some theoretical evidence for this. *So interesting!*

Improve cross-validation

- Even better: *repeated* cross-validation

Example:

10-fold cross-validation is repeated 10 times and results are averaged; and we adopt the union of rules as the new set of rules.

A particular form of cross-validation

- x-fold cross-validation: (N-N/x ; N/x)
- If x = N, what happens?
- We get:

(N-1; 1)

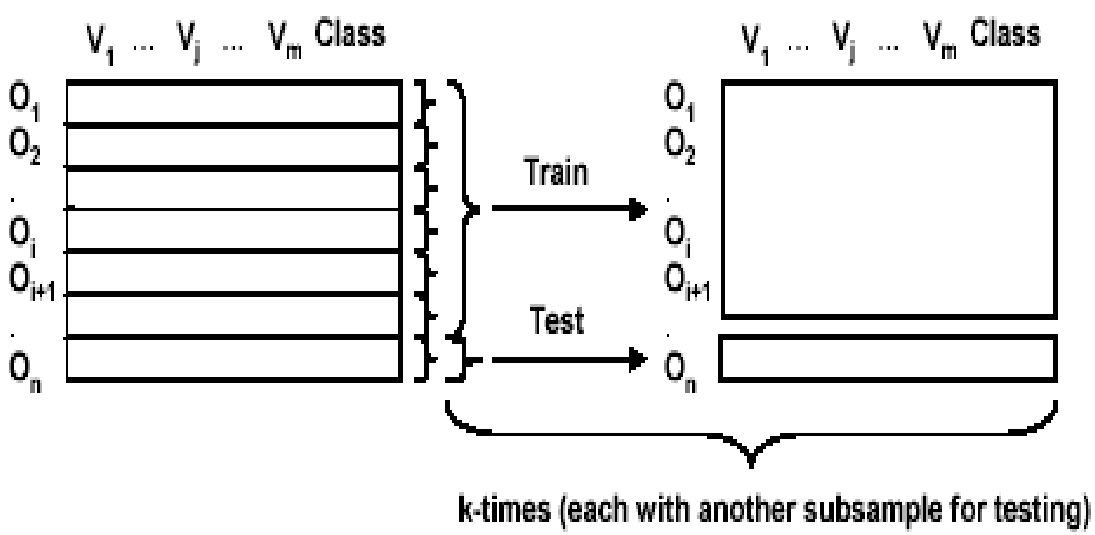
It is called “leave –one –out”

Leave-one-out (N-1 ; 1)

Cross-Validation (for moderated sample sizes) → Sampling without replacemen

- Dividing the given data set into ***m* subsamples of equal size**
- Each subsample is tested by using a model generated from the remaining *(m-1)* subsamples

→ **Leave-One-Out**: m = Number of objects



Leave-one-out (N-1 ; 1)

- Leave-one-out is a particular form of cross-validation:

  we set number of folds to number of training instances, i.e. x= N.

For n instances we build classifier

(repeat the training - testing) n times

Leave-one-out Procedure

- Let C(i) be the classifier (rules, patterns) built on all data except record x_i
- Evaluate C(i) on x_i, and determine if it is correct or in error
- Repeat for all i=1,2,…,n.
- The total error is the proportion of all the incorrectly classified x_i
- The final set of rules (patters) is a union of all rules obtained in the process.

Leave-one-out (N-1 ; 1)

- Make best use of the data
- Involves no random sub-sampling
- Stratification is not possible
- Very computationally expensive
- MOST commonly used