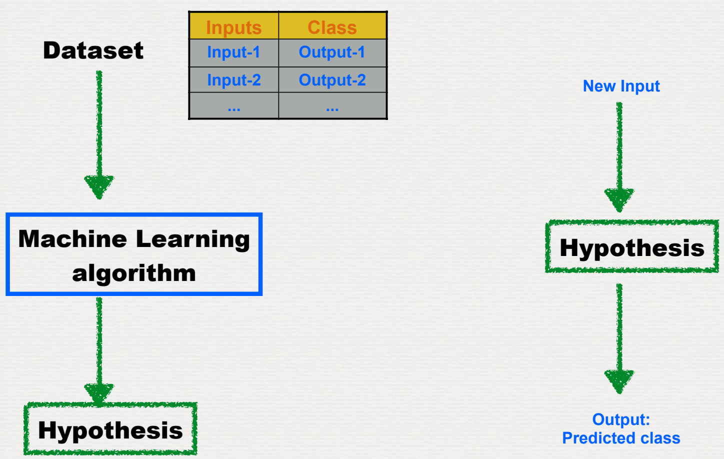
Sistemas Inteligentes
Intelligent Systems

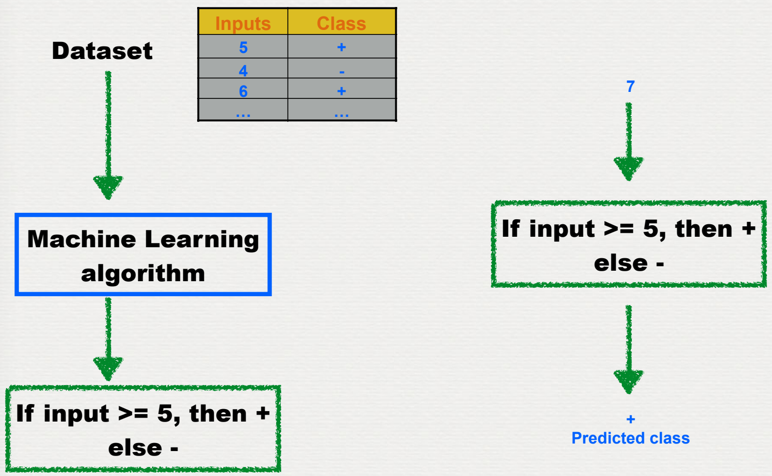
Tema 2. Aprendizaje Automático 3 (evaluación)
L2 Machine Learning 3 (evaluation)

Antonio Bahamonde
Departamento de informática
Universidad de Oviedo en Gijón

MACHINE LEARNING IN A SCHEMA



MACHINE LEARNING IN A SCHEMA



Evaluation: the key to success

- How predictive is the model we 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 lots 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

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

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
- **Rewriting** error: error rate obtained from training data
- **Rewriting** error is 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 can't be used for parameter tuning!
- Proper procedure uses three sets: training data, validation data, and test data
  - Validation data is used to optimize parameters

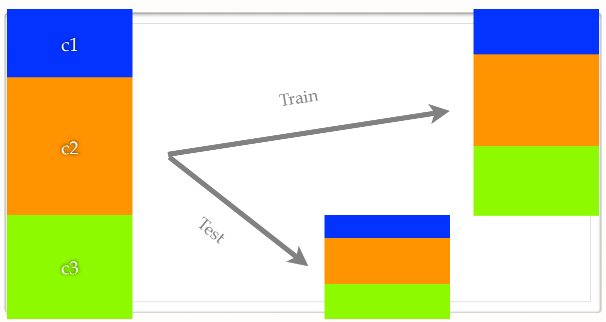
Making the most of the data

- **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
- 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!

Holdout estimation

- What to do if the amount of data is limited?
- The holdout method reserves a certain amount for testing and uses the remainder for training
  - 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**

STRATIFIED HOLDOUT



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 stratification)
  - 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?

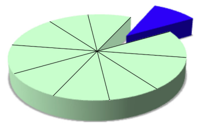
Cross Validation

- Can we improve upon repeated holdout?
(i.e. reduce variance)
- Cross-validation
- Stratified cross-validation

Cross Validation

10-fold cross-validation

- Divide dataset into 10 parts (folds)
- Hold out each part in turn
- Average the results
- Each data point used once for testing, 9 times for training

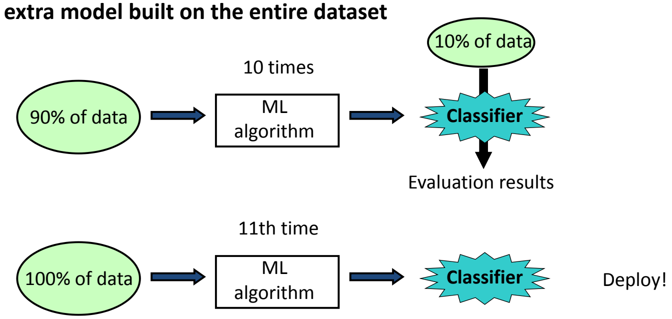


Stratified cross-validation

- Ensure that each fold has the right proportion of each class value

Cross Validation

After cross-validation, Weka outputs an extra model built on the entire dataset



Cross Validation

- Cross-validation better than repeated holdout
- Stratified is even better
- With 10-fold cross-validation, Weka invokes the learning algorithm 11 times
- **Practical rule of thumb:**
- Lots of data? – use percentage split
- Else stratified 10-fold cross-validation

Cross Validation

Is cross-validation really better than repeated holdout?

- **Diabetes** dataset
- Baseline accuracy (**rules > ZeroR**): 65.1%
- **trees > J48**
- 10-fold cross-validation 73.8%
- … with different random number seed

1	2	3	4	5	6	7	8	9	10
73.8	75.0	75.5	75.5	74.4	75.6	73.6	74.0	74.5	73.0

Cross Validation

Sample mean $\overline{x} = \frac{\Sigma x_i}{n}$

Variance $\sigma^2 = \frac{\Sigma (x_i - \overline{x})^2}{n-1}$

Standard deviation σ

holdout (10%)	cross-validation (10-fold)
75.3	73.8
77.9	75.0
80.5	75.5
74.0	75.5
71.4	74.4
70.1	75.6
79.2	73.6
71.4	74.0
80.5	74.5
67.5	73.0
$\overline{x}$ = 74.8	$\overline{x}$ = 74.5
σ = 4.6	σ = 0.9

Cross Validation

- Why 10-fold? E.g. 20-fold: 75.1%

- Cross-validation really is better than repeated holdout
- It reduces the variance of the estimate

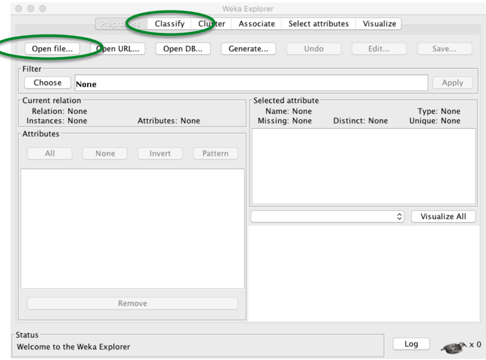
Leave-One-Out cross-validation

- Leave-One-Out: a particular form of 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
  - (exception: NN)

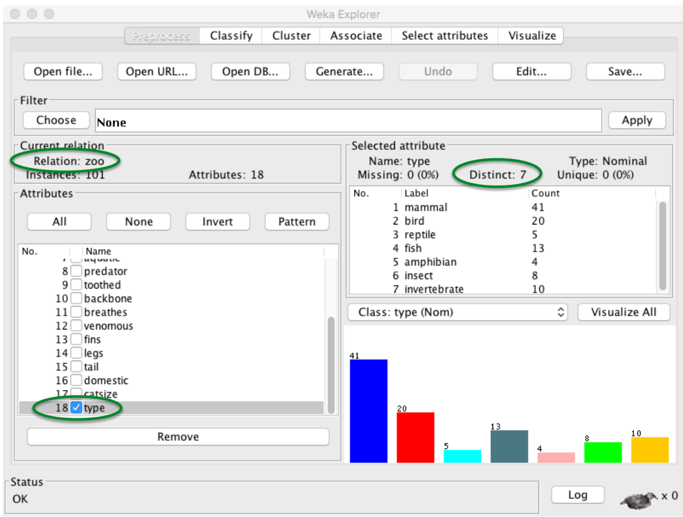
LEAVE-ONE-OUT (LOO)

- Its most important disadvantage:
  - It is not possible to make a stratified version
  - Let us suppose that a dataset has exactly 50 examples of one class, and 50 of other class. Then Zero R has an expected error rate of 50%. However, the loo estimation returns 100%

Weka



Weka



Weka