Lecture 6: Assessment of machine (supervised) learning systems

Universitat de les Illes Balears
Departament de Ciències Matemàtiques i Informàtica

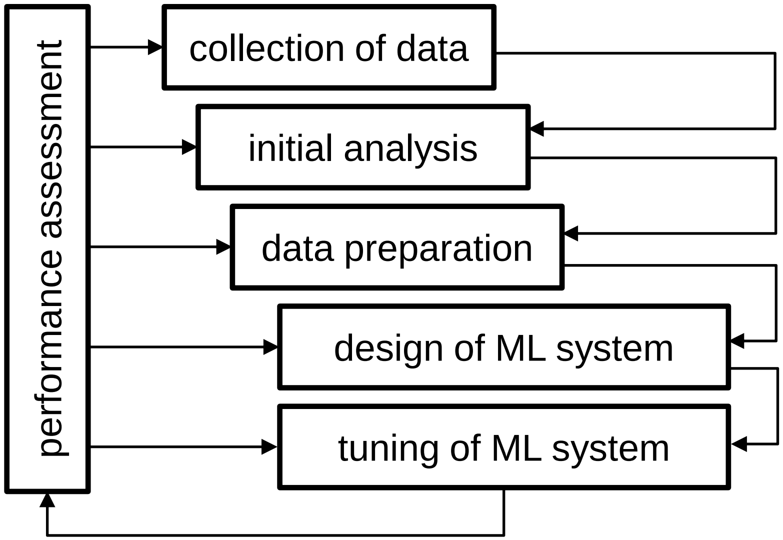
11752 Aprendizaje Automático
Máster Universitario
en Sistemas Inteligentes

Alberto ORTIZ RODRÍGUEZ

Contents

Introduction

- The assessment of ML systems involves several aspects of its performance and may take the designer back to any of the developing stages



- A first evaluation setting, splits the dataset into two subsets:
  - **training dataset**
    - used to build the classifier/regressor
  - **test dataset**
    - to check the behaviour of the classifier with unseen examples
  - both include the corresponding **ground truth**

Introduction

- Performance assessment of ML systems should be ensured to be unbiased

  $\Rightarrow$ **cross-validation** techniques

- ML systems evaluation tools are useful not only for assessing the performance of the system, but also
  - for **debugging** purposes, to figure out what is not working and make the necessary adjustments
  - to **compare among different ML techniques/methods**

Contents

Bias-variance tradeoff

- During ML systems development, the main concern is their **generalization error**
- The **bias-variance decomposition** (BVD) is a way of analyzing the expected generalization error of an ML system
  - Let us assume a **training dataset** D = $\{(x_1,y_1), (x_2,y_2), \ldots, (x_N,y_N)\}$
    - y_i can be either a continuous value (regression) or a class label (classification)
  - We also assume $y(x) = f(x) + \varepsilon$, where ε is white noise, i.e. $E[\varepsilon] = 0$ and $Var[\varepsilon] = \sigma^2$
  - We are intent to find an approximation $f_D(x)$ of the true function f(x) by means of learning
    - To find the approximation we make use of **D** and a certain **ML model** $\mathcal{M}$, e.g.

$$f_D \text{ is such that } \sum_{i=1}^{N}(f_D(x_i) - y_i)^2 \text{ is minimal for } D$$

  - It turns out that the expected error on the **unseen test dataset** {(x,y)} comprises three terms:

$$\mathrm{E}_{x,y,D}\left[(f_D(x) - y(x))^2\right] = \mathrm{Bias}^2\left[\mathcal{M}\right] + \mathrm{Var}\left[\mathcal{M}\right] + \sigma^2$$

  where $\mathrm{Bias}^2[\mathcal{M}] = \mathrm{E}_x\left[\left(f(x) - \bar{f}\left(x\right)\right)^2\right]$ and $\mathrm{Var}[\mathcal{M}] = \mathrm{E}_{x,D}\left[\left(f_D(x) - \bar{f}\left(x\right)\right)^2\right]$
  or, in other words:

  generalization error = **bias** error term + **variance** error term + **irreducible** error term
  - All terms are non-negative, hence BVD becomes a sort of **lower bound** of the expected error on unseen samples and datasets
    - we are aware we cannot do it perfectly, since the y_i are affected by an irreducible error ε, but
    - we can try to reduce/compensate the bias and variance error terms

- **proof**
  - Some notation first:

$$\text{Relation between } x \text{ and } y\text{: } y(x) = f(x) + \varepsilon\,, \mathrm{E}\left[\varepsilon\right] = 0, \mathrm{Var}[\varepsilon] = \sigma^2$$

$$\text{Expected output: } \bar{y}(x) = \mathrm{E}_{y|x}\left[y\right] = \int_y y(x)\,\mathrm{p}(y|x)\,\mathrm{d}y = f(x)$$

$$\text{Expected classifier: } \bar{f}\,(x) = \mathrm{E}_D\left[f_D(x)\right] = \int_D f_D(x)\,\mathrm{p}(D)\,\mathrm{d}D$$

$$\text{Expected test error: } \mathrm{E}_{x,y,D}\left[\left(f_D(x) - y(x)\right)^2\right] = \int_D \int_x \int_y \left(f_D(x) - y(x)\right)^2 \mathrm{p}(x,y)\mathrm{p}(D)\,\mathrm{d}x\mathrm{d}y\mathrm{d}D$$

  - The expected test error can now be stated as follows:

$$\begin{aligned}\mathrm{E}_{x,y,D}\left[\left(f_D(x) - y(x)\right)^2\right] &= \mathrm{E}_{x,y,D}\left[\left(f_D(x) - \bar{f}\,(x) + \bar{f}\,(x) - y(x)\right)^2\right] \\ &= {\color{red}\mathrm{E}_{x,D}\left[(f_D(x) - \bar{f}\,(x))^2\right]} + \mathrm{E}_{x,y}\left[\left(\bar{f}\,(x) - y(x)\right)^2\right] \\ &\quad + 2\mathrm{E}_{x,y,D}\left[\left(f_D(x) - \bar{f}\,(x)\right)\left(\bar{f}\,(x) - y(x)\right)\right]\end{aligned}$$

  - The third term can be shown to vanish:

$$\begin{aligned}\mathrm{E}_{x,y,D}\left[\left(f_D(x) - \bar{f}\,(x)\right)\left(\bar{f}\,(x) - y(x)\right)\right] &= \mathrm{E}_{x,y}\left[\mathrm{E}_D\left[f_D(x) - \bar{f}\,(x)\right]\left(\bar{f}\,(x) - y(x)\right)\right] \\ &= \mathrm{E}_{x,y}\left[\left(\mathrm{E}_D\left[f_D(x)\right] - \bar{f}\,(x)\right)\left(\bar{f}\,(x) - y(x)\right)\right] \\ &= \mathrm{E}_{x,y}\left[\left(\bar{f}\,(x) - \bar{f}\,(x)\right)\left(\bar{f}\,(x) - y(x)\right)\right] = 0\end{aligned}$$

- **proof** (contd.)
  - On the other side, the second term can be also stated as follows:

$$\begin{aligned}
\mathrm{E}_{x,y}\left[\left(\bar{f}(x) - y(x)\right)^2\right] &= \mathrm{E}_{x,y}\left[\left(\bar{f}(x) - f(x) + f(x) - y(x)\right)^2\right] \\
&= \mathrm{E}_{x}\left[\left(\bar{f}(x) - f(x)\right)^2\right] + \mathrm{E}_{x,y}\left[\left(f(x) - y(x)\right)^2\right] \\
&+ 2\mathrm{E}_{x,y}\left[\left(\bar{f}(x) - f(x)\right)\left(f(x) - y(x)\right)\right]
\end{aligned}$$

  - This last term can also be shown to vanish:

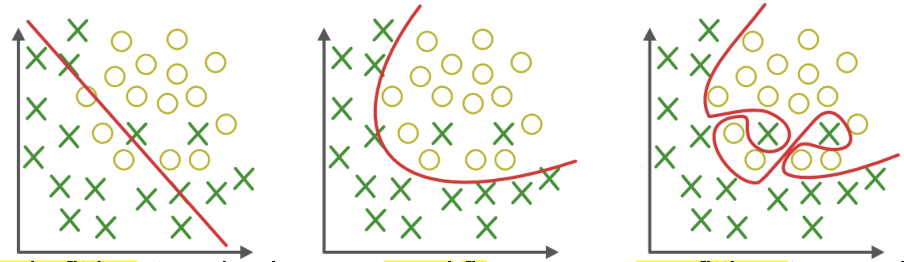
$$\begin{aligned}
\mathrm{E}_{x,y}\left[\left(\bar{f}(x) - f(x)\right)\left(f(x) - y(x)\right)\right] &= \mathrm{E}_{x}\left[\left(\bar{f}(x) - f(x)\right)\mathrm{E}_{y|x}\left[f(x) - y(x)\right]\right] \\
&= \mathrm{E}_{x}\left[\left(\bar{f}(x) - f(x)\right)\left(f(x) - \mathrm{E}_{y|x}\left[y\right]\right)\right] \\
&= \mathrm{E}_{x}\left[\left(\bar{f}(x) - f(x)\right)\left(f(x) - f(x)\right)\right] = 0
\end{aligned}$$

  - Finally, we obtain the decomposition as follows:

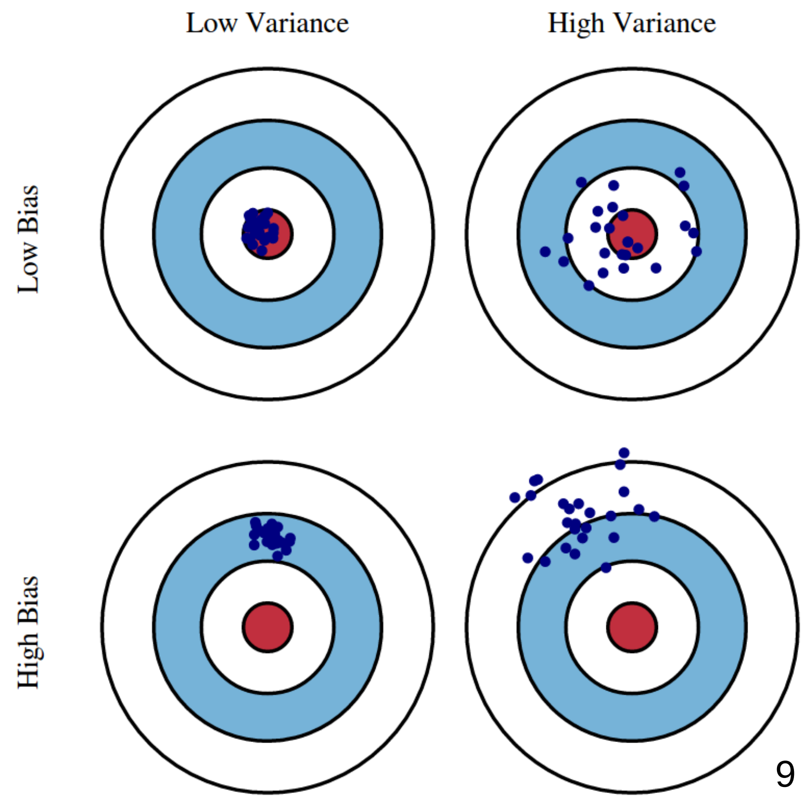
$$\begin{aligned}
\mathrm{E}_{x,y,D}\left[\left(f_D(x) - \bar{f}(x)\right)\left(\bar{f}(x) - y(x)\right)\right] &= \mathrm{E}_{x}\left[\left(f(x) - \bar{f}(x)\right)^2\right] && \text{(bias term)} \\
&+ \mathrm{E}_{x,D}\left[\left(f_D(x) - \bar{f}(x)\right)^2\right] && \text{(variance term)} \\
&+ \mathrm{E}_{x,y}\left[\left(f(x) - y(x)\right)^2\right] && \text{(irreducible noise term)}
\end{aligned}$$

Bias-variance tradeoff

- An effective way to get a clearer idea of bias and variance errors is through a visual representation
- A more general view:
  - the inner red circle represents good models
  - every point ("dart") represents one prediction
  - **low variance – low bias**
    - good understanding of data patterns
    - predictions hit the bull's eye
  - **high variance – low bias**
    - learn proper patterns and perform decently on average
    - sensitive to the data it is trained on and predictions keep fluctuating
  - **low variance – high bias**
    - consistent predictions but proper patterns have not been learnt correctly
  - **high variance – high bias**
    - proper patterns have not been learnt properly
    - extremely sensitive to data noise and outliers leading to highly fluctuating predictions

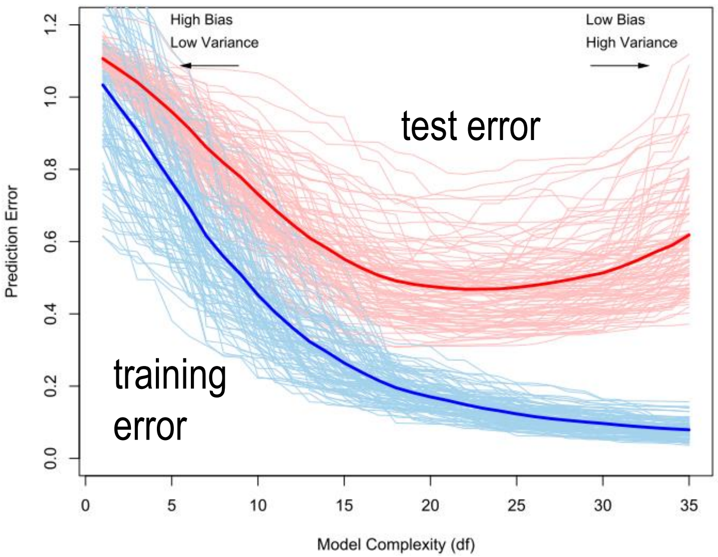


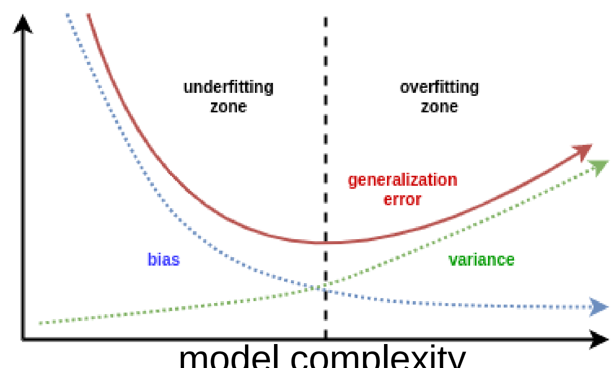
underfitting: too simple to explain data (high bias) | good fit | overfitting: too good to be true (high variance)



Bias-variance tradeoff

- Summing up:
  - **Bias**
    - It is due to wrong assumptions, either by the designer or by the model
    - A high-bias model is most likely to **underfit** the training data
  - **Variance**
    - It is due to the model's **excessive sensitivity** to small variations in the training data
    - A model with many degrees of freedom, e.g. a high-degree polynomial, is likely to have high variance and thus to **overfit** the training data
  - **Irreducible error**
    - It is due to the **noisiness** of the data itself
    - the only way to reduce this error is to clean up the data, i.e. fix a broken sensor
- **Increasing a model's complexity** typically increases its variance and reduces its bias
- **Reducing a model's complexity** usually increases its bias and reduces its variance
- This is why it is called the **bias-variance tradeoff** (also BV dilemma)





Contents

Confusion matrix

- Many **performance metrics** can be calculated from the **confusion matrix**, e.g. for two classes, having defined first which is the positive class

		predicted class		
		positive	**negative**	
true	**positive**	TP	FN	$\mathrm{Po} = TP + FN$
class	**negative**	FP	TN	$\mathrm{Ne} = FP + TN$
		$\widehat{\mathrm{Po}} = TP + FP$	$\widehat{\mathrm{Ne}} = FN + TN$	

accuracy (A)	$\frac{TP+TN}{TP+TN+FP+FN}$	**error rate** (E)	$1 - \mathrm{A} = \frac{FP+FN}{TP+TN+FP+FN}$
false positive rate (FPR)	$\frac{FP}{\mathrm{Ne}} = \frac{FP}{FP+TN}$	**true positive rate** (TPR)	$\frac{TP}{\mathrm{Po}} = \frac{TP}{TP+FN}$
precision (P)	$\frac{TP}{\widehat{\mathrm{Po}}} = \frac{TP}{TP+FP}$	**recall** (R)	$\frac{TP}{\mathrm{Po}} = TPR$
specificity	$\frac{TN}{\mathrm{Ne}} = 1 - FPR$	F_1**-score**	$\frac{2}{\frac{1}{P}+\frac{1}{R}} = \frac{2PR}{P+R}$

FPR is also known as **false alarm rate**

TPR is also knwon as **sensitivity**

- **F_1 -score** combines in a single metric P and R, by means of their **harmonic mean**
  - the regular mean treats all values equally, the harmonic mean weighs more low values
  - a high F_1 score (which is better) results only if both P and R are high
  - also known as Sorensen-Dice coefficient or **Dice Similarity Coefficient** (DSC)

- F_1 is a particular case of the **F_β -score**:

$$F_\beta = (1+\beta^2)\frac{P\,R}{\beta^2 P + R} = \frac{(1+\beta^2)TP}{(1+\beta^2)TP + \beta^2 FN + FP}$$

$$F_1 = \frac{2TP}{2TP + FN + FP}, \quad F_2 = \frac{5TP}{5TP + 4FN + FP}, \quad F_{0.5} = \frac{1.25\,TP}{1.25\,TP + 0.25FN + FP}$$

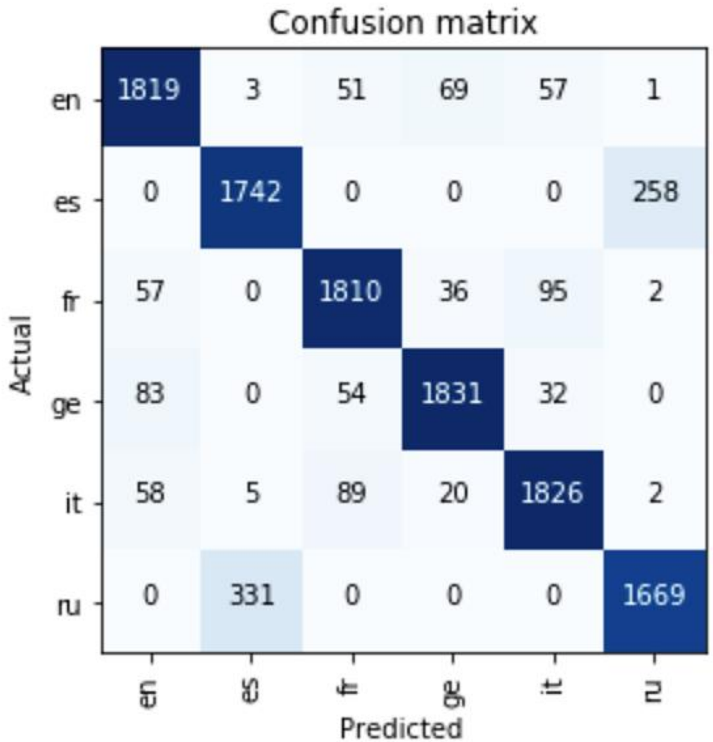
  - $\beta = 1$, F_1 -score, which weighs equally R and P: same emphasis on FN and FP
  - $\beta = 2$, **F_2 -score**, which weighs R lower than P: effect of FP is less noticeable
  - $\beta = 0.5$, **$F_{0.5}$ -score**, which weighs R higher than P: effect of FN is less noticeable

$$P = \frac{TP}{\widehat{\mathrm{Po}}} = \frac{TP}{TP + FP}$$

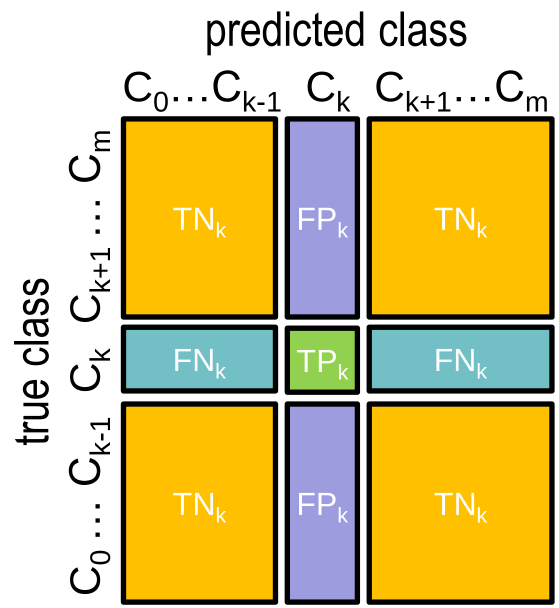
$$R = \frac{TP}{\mathrm{Po}} = \frac{TP}{TP + FN}$$

Confusion matrix

- The confusion matrix can be generalized for **M-class problems**



TP, TN, FP, FN are calculated according to **one versus all (OvA) classification**



- Global metrics can be calculated following two approaches:
  - **micro-averages**: from individual TP, TN, FP, FN
    - each prediction is weighed equally
  - **macro-averages**: average scores for each class
    - each class is weighed equally

$$\text{e.g.}\;\; P_{\text{micro}} = \frac{TP_1 + \cdots + TP_m}{TP_1 + \cdots + TP_m + FP_1 + \cdots + FP_m}$$

$$\text{e.g.}\;\; P_{\text{macro}} = \frac{P_1 + \cdots + P_m}{m}\,, \qquad P_i = \frac{TP_i}{TP_i + FP_i}$$

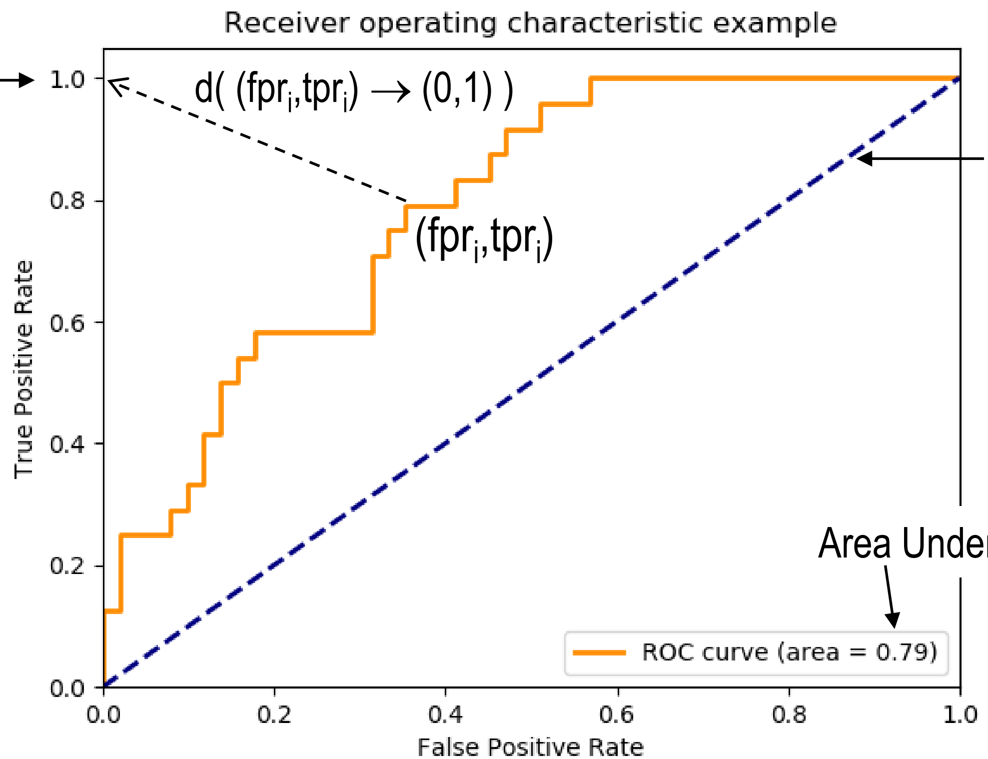
Contents

- Introduction
- Bias-variance tradeoff
- Confusion matrix and performance metrics
- ROC curves
- Cross-validation techniques

- **Receiver Operating Characteristic** curve (concept from early Radar days)
  - set of (FPR, TPR) points obtained by varying the algorithm's parameters
    - every (FPR, TPR) point is 1 classifier configuration

(FPR=0, TPR=1) perfect classifier

$$FPR = \frac{FP}{\mathrm{Ne}} = \frac{FP}{FP+TN}$$
$$TPR = \frac{TP}{\mathrm{Po}} = \frac{TP}{TP+FN}$$



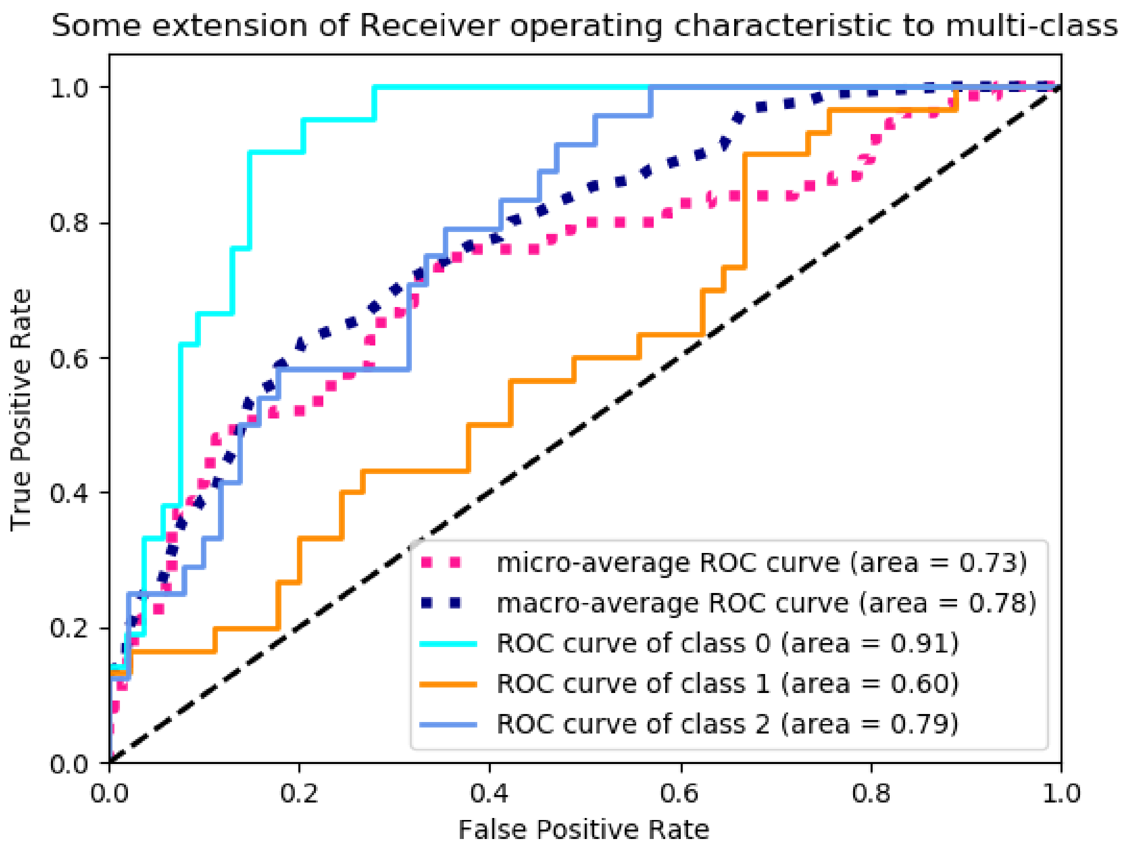
random guessing: FPR = TPR ⇒ FP vs Negatives = TP vs Positives (no-discrimination line)

Area Under the Curve (AUC)

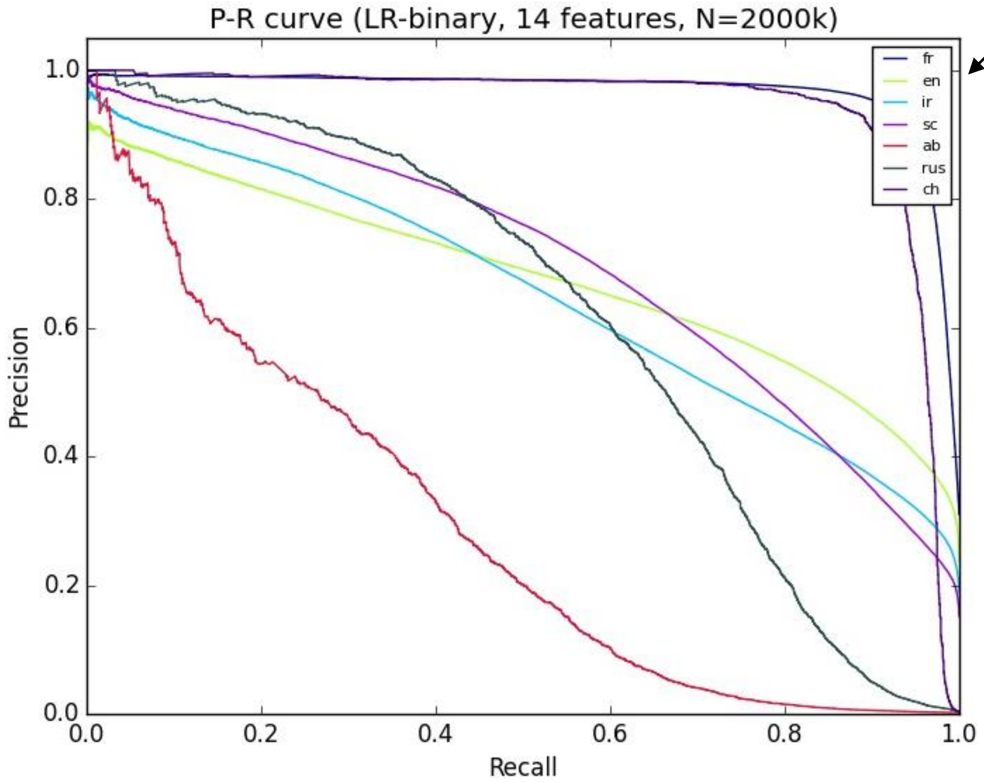
- can also be shown as sensitivity (TPR) vs 1 – specificity (FPR)
  - **Area Under the Curve** (AUC)
    - global measure of classifier performance, the higher the better

ROC curves

- **Receiver Operating Characteristic** curve (concept from early Radar days)
  – can also be calculated for **multi-class problems**



- Other usual curves for classifier performance characterization are the **Precision-Recall** curves



(R=1, P=1)
perfect classifier

$$P = \frac{TP}{\widehat{\text{Po}}} = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{\text{Po}} = \frac{TP}{TP + FN}$$

  – Note: TN are not accounted for in this curve, it is the kind of problem where negatives are not as relevant as positives, e.g. inspection systems

Contents

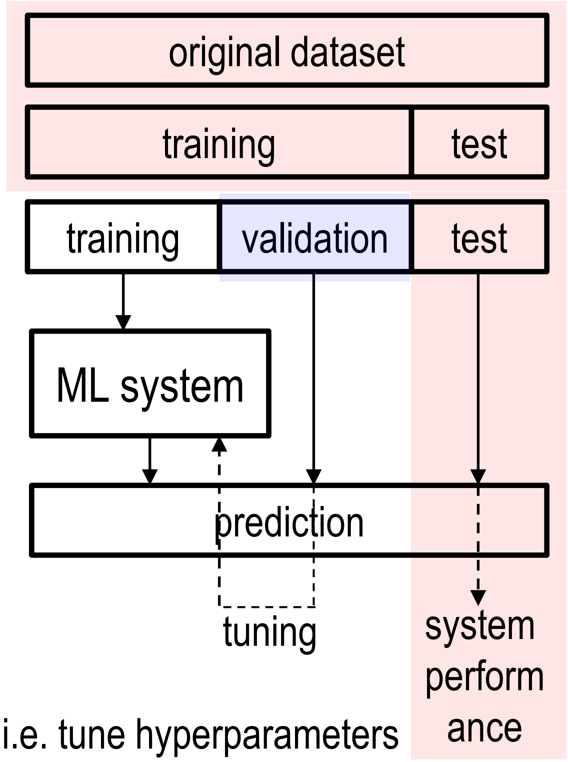
- Introduction
- Bias-variance tradeoff
- Confusion matrix and performance metrics
- ROC curves
- Cross-validation techniques

Cross-validation techniques

- Cross-validation techniques can provide performance estimation values with low bias:
  - holdout cross validation
  - n-fold cross validation
  - stratified n-fold cross validation
  - leave-one-out cross validation (LOOCV)
  - nested cross validation

Cross-validation techniques

- **Holdout cross validation**

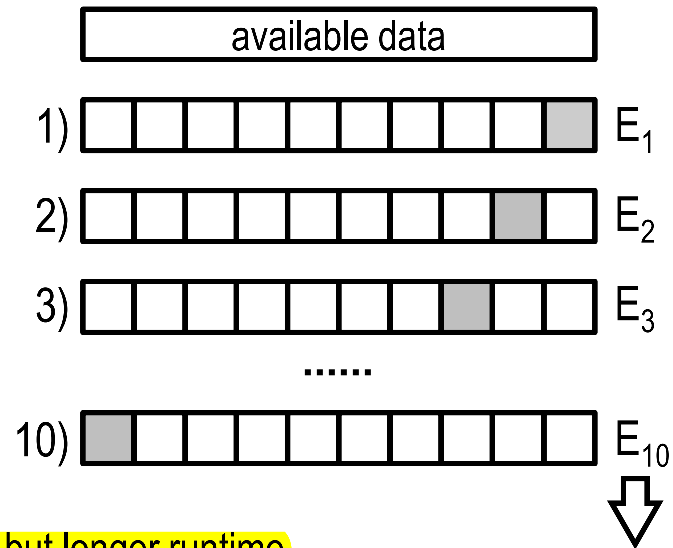


  - Simplest kind of cross validation
  - The data set is initially split into two sets: the **training set** and the **test set**
    1. ML system is built using the training set only
    2. ML system is asked to predict the output values for the data in the “unseen” test set
  - The accumulated errors give the **test error**, used to evaluate the model
  - To tune the system appropriately, we can split the training set in a **training subset** and a **validation subset**
    - The validation subset can be employed for model selection, i.e. tune hyperparameters
      - train the system
      - repeatedly evaluate it using the validation subset using different settings
  - The evaluation may depend heavily on which data points end up in the training set and which end up in the test set, and thus the evaluation may be significantly different depending on how the splitting is made

→ Maybe **high variance** in performance estimation if the test is repeated (as splitting is random)

Cross-validation techniques

- **n-fold cross validation**
  - The available data is randomly split into n folds without replacement:
    - n-1 folds are used for training
    - the remaining fold is used for test
  - The procedure is repeated n times, choosing a different fold for testing each iteration
  - A global performance measure is obtained by averaging the individual measurements
    - lower variance estimate than the holdout method
  - In most cases, n = 5 or 10
    - n large ⇒ more data for training ⇒ lower bias in E but longer runtime



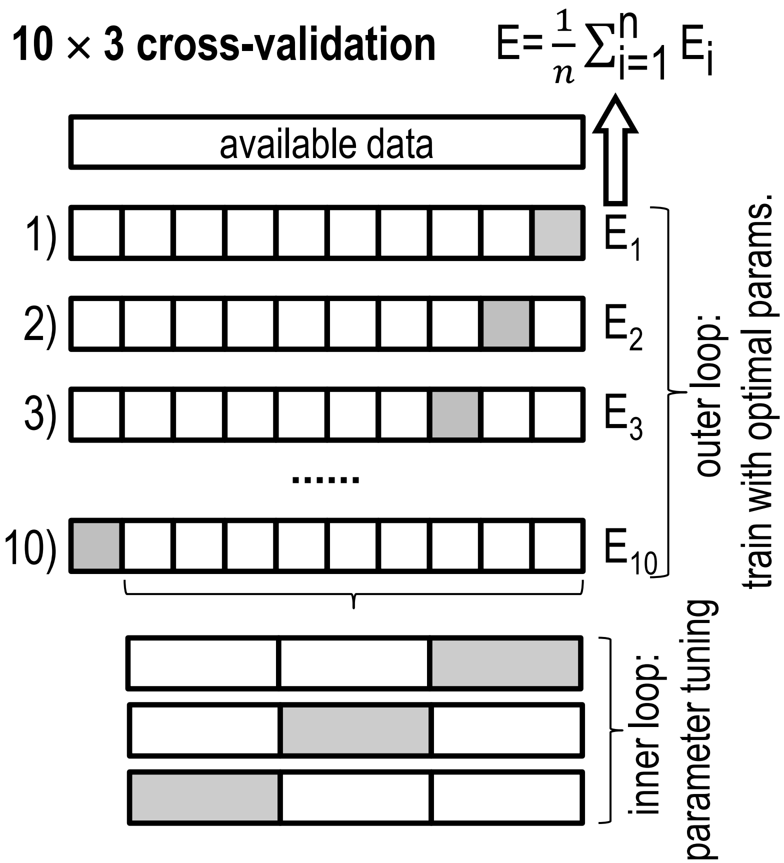
$$E = \frac{1}{n} \sum_{i=1}^{n} E_i$$

  - **Stratified n-fold cross validation**
    - Class proportions are preserved in each fold
    - Better performance estimates as for bias and variance

  - **Leave-one-out cross validation** (LOOCV)
    - n = N, the size of the dataset; hence, at each iteration, the test set comprises one single sample
    - recommended for specific cases, e.g. very small datasets

Cross-validation techniques

- **Nested cross validation**
  - Outer n-fold: performance estimation
    the available data is randomly split into **n folds** without replacement:
    - n-1 folds are used for training
    - the remaining fold is used for test
    - The procedure is repeated n times, choosing a different fold for testing each iteration
  - Inner m-fold: model selection (hyperp. tuning)
    the set of training folds is split into **m folds** leaving one for validation
  - A global performance measure is obtained by averaging the individual measurements
  - This measure gives a good estimate of what to expect from unseen data



Lecture 6: Assessment of machine (supervised) learning systems

Universitat de les Illes Balears
Departament de Ciències Matemàtiques i Informàtica

11752 Aprendizaje Automático
Máster Universitario en Sistemas Inteligentes

Alberto ORTIZ RODRÍGUEZ