Chapter ML:II (continued)

II. Machine Learning Basics

- On Data
- Regression
- Concept Learning: Search in Hypothesis Space
- Concept Learning: Search in Version Space
- Measuring Performance

Definition 8 (True Misclassification Rate)

Let X be a feature space with a finite number of elements. Moreover, let C be a set of classes, let $y : X \rightarrow C$ be a classifier, and let c be the target concept to be learned. Then the true misclassification rate, denoted as *Err*$^*(y)$, is defined as follows:

$$\textit{Err}^*(y) = \frac{|\{\mathbf{x} \in X : c(\mathbf{x}) \neq y(\mathbf{x})\}|}{|X|}$$

Measuring Performance

True Misclassification Rate

Definition 8 (True Misclassification Rate)

Let X be a feature space with a finite number of elements. Moreover, let C be a set of classes, let $y : X \to C$ be a classifier, and let c be the target concept to be learned. Then the true misclassification rate, denoted as *Err*$^*(y)$, is defined as follows:

$$\textit{Err}^*(y) = \frac{|\{\mathbf{x} \in X : c(\mathbf{x}) \neq y(\mathbf{x})\}|}{|X|}$$

Problem:

- Usually c is unknown.

Solution:

- Estimation of *Err*$^*(y)$ with *Err*(y, D_s), i.e., by evaluating y on a sample $D_s \subseteq D$. Recall that for the feature vectors in D the target concept c is known.

Remarks:

- The English word “rate” can be used to denote both the mathematical concept of a flow quantity (a change of a quantity per time unit) as well as the mathematical concept of a *portion*, a *percentage*, or a *ratio*, which has a stationary (= time-independent) semantics. This latter semantics is meant here when talking about the misclassification rate.

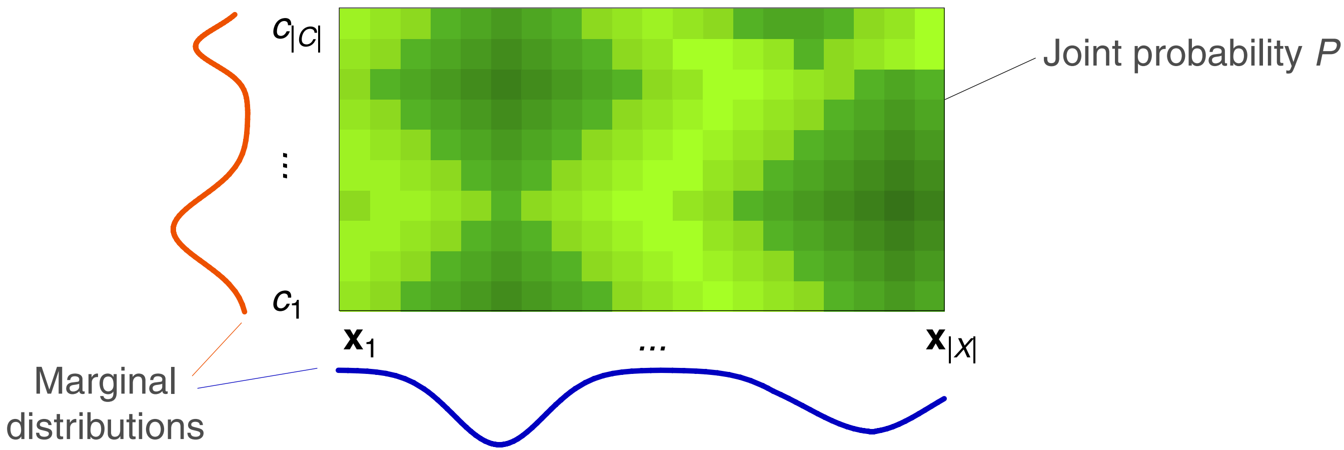
- Unfortunately, the German word “Rate” is often (mis)used to denote the mathematical concept of a portion, a percentage, or a ratio. Taking a precise mathematical standpoint, the correct German words are “Anteil” or “Quote”. I.e., a semantically correct translation of misclassification rate is “Missklassifikationsanteil”, and not “Missklassifikationsrate”.

Measuring Performance

True Misclassification Rate (continued)

Probabilistic foundation [ML:IV Probability Basics] :

- Let X and C be defined as before. Moreover, let P be a probability measure on $X \times C$. Then $P(\mathbf{x}, c)$ (precisely: $P(\mathcal{H} = \mathbf{x}, \mathcal{C} = c)$) denotes the probability that feature vector $\mathbf{x} \in X$ belongs to class $c \in C$. Illustration:

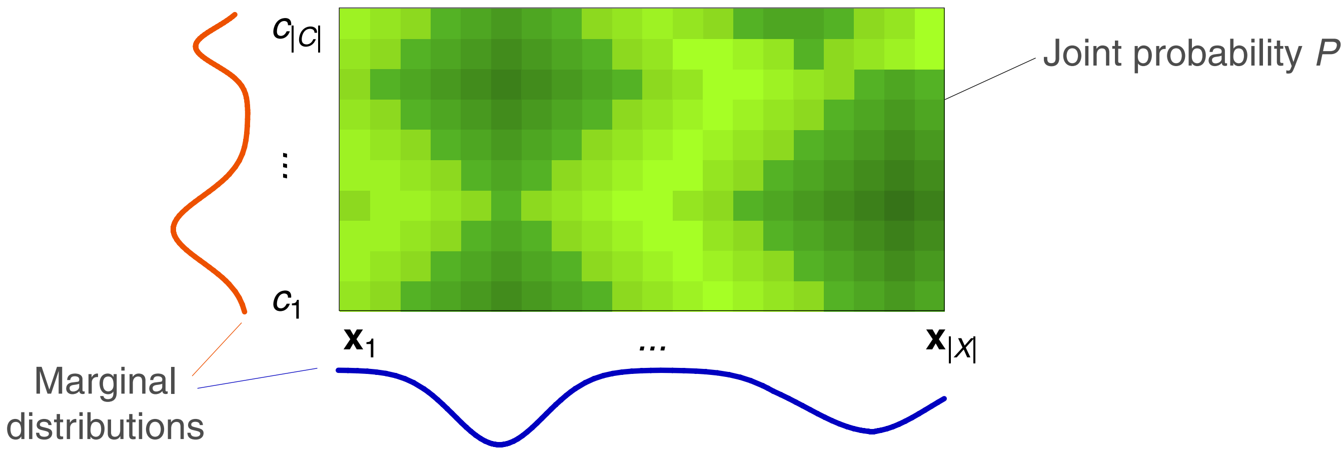


Measuring Performance

True Misclassification Rate (continued)

Probabilistic foundation [ML:IV Probability Basics] :

- Let X and C be defined as before. Moreover, let P be a probability measure on $X \times C$. Then $P(\mathbf{x}, c)$ (precisely: $P(\mathcal{H} = \mathbf{x}, \mathcal{C} = c)$) denotes the probability that feature vector $\mathbf{x} \in X$ belongs to class $c \in C$. Illustration:



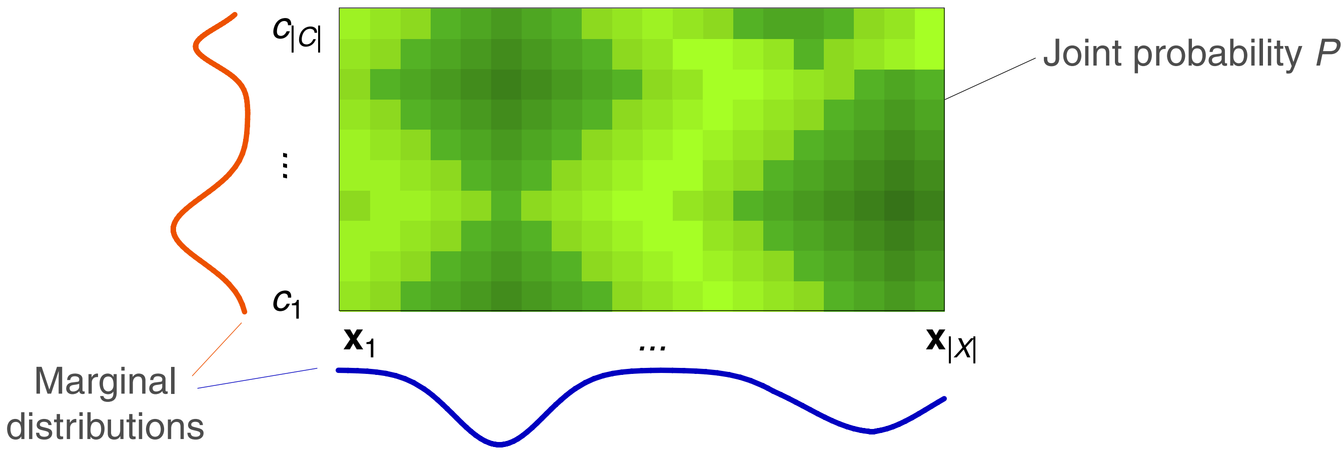
- $\textit{Err}^*(y) = \sum_{\mathbf{x} \in X} \sum_{c \in C} P(\mathbf{x}, c) \cdot I(y(\mathbf{x}), c), \quad$ with $I(y(\mathbf{x}), c) = \begin{cases} 0 & \text{if } y(\mathbf{x}) = c \\ 1 & \text{otherwise} \end{cases}$

Measuring Performance

True Misclassification Rate (continued)

Probabilistic foundation [ML:IV Probability Basics] :

- Let X and C be defined as before. Moreover, let P be a probability measure on $X \times C$. Then $P(\mathbf{x}, c)$ (precisely: $P(\mathcal{H} = \mathbf{x}, \mathcal{C} = c)$) denotes the probability that feature vector $\mathbf{x} \in X$ belongs to class $c \in C$. Illustration:



- $Err^*(y) = \sum_{\mathbf{x} \in X} \sum_{c \in C} P(\mathbf{x}, c) \cdot I(y(\mathbf{x}), c)$, with $I(y(\mathbf{x}), c) = \begin{cases} 0 & \text{if } y(\mathbf{x}) = c \\ 1 & \text{otherwise} \end{cases}$

- $D = \{(\mathbf{x}_1, c_1)), \ldots, (\mathbf{x}_n, c_n))\} \subseteq X \times C$ is a set of examples whose elements are drawn independently and according to the same P.

Remarks:

- Let A and B denote two events, e.g., $A =$ "$\mathcal{H} = \mathbf{x}$" and $B =$ "$\mathcal{C} = c$". Then the following expressions are syntactic variants, i.e., they are semantically equivalent: $P(A, B)$, $P(A$ and $B)$, $P(A \wedge B)$
- $\mathcal{H}$ and $\mathcal{C}$ are random variables with domains X and C respectively.
- The function $c(\mathbf{x})$ has been modeled as random variable, $\mathcal{C}$, since in the real world the classification of a feature vector $\mathbf{x}$ may not be deterministic but the result of a random process. Keyword: label noise.
- The elements in D are considered as random variables that are both independent of each other and identically distributed. This property of a set of random variables is abbreviated with "i.i.d."

Measuring Performance
Training Error

- $D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))\} \subseteq X \times C$ is a set of examples.
- $D_{tr} = D$ is the training set.
- $y : X \to C$ is a classifier learned on the basis of D_{tr}.

Training error = misclassification rate with respect to D_{tr} :

$$\mathit{Err}(y, D_{tr}) = \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D_{tr} : c(\mathbf{x}) \neq y(\mathbf{x})\}|}{|D_{tr}|}$$

Measuring Performance

Training Error

- $D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))\} \subseteq X \times C$ is a set of examples.
- $D_{tr} = D$ is the training set.
- $y : X \to C$ is a classifier learned on the basis of D_{tr}.

Training error = misclassification rate with respect to D_{tr} :

$$\mathit{Err}(y, D_{tr}) = \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D_{tr} : c(\mathbf{x}) \neq y(\mathbf{x})\}|}{|D_{tr}|}$$

Problems:

- *Err*(y, D_{tr}) is based on examples that are also exploited to learn y.

➜ *Err*(y, D_{tr}) quantifies memorization but not the generalization capability of y.

➜ *Err*(y, D_{tr}) is an optimistic estimation, i.e., it is constantly lower compared to an application of y in the wild.

Measuring Performance

Holdout Estimation

- $D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))\} \subseteq X \times C$ is a set of examples.
- $D_{tr} \subset D$ is the training set.
- $y : X \to C$ is a classifier learned on the basis of D_{tr}.
- $D_{ts} \subset D$ with $D_{ts} \cap D_{tr} = \emptyset$ is a test set.

Holdout estimation = misclassification rate with respect to D_{ts} :

$$\mathit{Err}(y, D_{ts}) = \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D_{ts} : c(\mathbf{x}) \neq y(\mathbf{x})\}|}{|D_{ts}|}$$

Measuring Performance

Holdout Estimation

- $D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))\} \subseteq X \times C$ is a set of examples.
- $D_{tr} \subset D$ is the training set.
- $y : X \to C$ is a classifier learned on the basis of D_{tr}.
- $D_{ts} \subset D$ with $D_{ts} \cap D_{tr} = \emptyset$ is a test set.

Holdout estimation = misclassification rate with respect to D_{ts} :

$$\mathit{Err}(y, D_{ts}) = \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D_{ts} : c(\mathbf{x}) \neq y(\mathbf{x})\}|}{|D_{ts}|}$$

Requirements:

- D_{tr} and D_{ts} must be drawn i.i.d.
- D_{tr} and D_{ts} must have similar sizes.

Remarks:

- A typical value for splitting D into training set D_{tr} and test set D_{ts} is 2:1.
- When splitting D into D_{tr} and D_{ts} one has to ensure that the underlying distribution is maintained. Keywords: stratification, sample selection bias

Measuring Performance

Cross Validation: k-Fold

Improved cross validation for small sets D :

- Form k test sets by splitting D into disjoint sets $D_1, \ldots, D_k$ of similar size.
- For $i = 1, \ldots, k$ do:

  1. $y_i : X \to C$ is a classifier learned on the basis of $D \setminus D_i$
  2. $\mathit{Err}(y_i, D_i) = \dfrac{|\{(\mathbf{x}, c(\mathbf{x}) \in D_i : y_i(\mathbf{x}) \neq c(\mathbf{x})\}|}{|D_i|}$

Cross-validated misclassification rate:

$$\mathit{Err}_{cv}(y, D, k) = \frac{1}{k} \sum_{i=1}^{k} \mathit{Err}(y_i, D_i)$$

Remarks:

- Rationale: For large k the set $D \setminus D_i$ is of similar size as D. Hence *Err*$^*(y_i)$ is close to *Err*$^*(y)$, where y is the classifier learned on the basis of D.
- For the construction of tree classifiers, tenfold cross-validation has been reported to give good results. [Breiman]

Measuring Performance

Cross Validation: Leave One Out

Special case of cross validation with $k = n$:

- Determine the cross-validated misclassification rate for $D \setminus D_i$ where $D_i = \{(\mathbf{x}_i, c(\mathbf{x}_i))\},\ i \in \{1, \ldots, n\}$.

Measuring Performance

Cross Validation: Leave One Out

Special case of cross validation with $k = n$:

- Determine the cross-validated misclassification rate for $D \setminus D_i$ where $D_i = \{(\mathbf{x}_i, c(\mathbf{x}_i))\},\ i \in \{1, \ldots, n\}$.

Problems:

- High computational effort if D is large.
- Singleton test sets ($|D_i| = 1$) are never stratified since they contain a single class only.

Measuring Performance
Bootstrapping

Multiple exploitation of D :

- For $i = 1, \ldots, k$ do:
  1. Form training set D_i by drawing n examples from D *with replacement*.
  2. $y_i : X \to C$ is a classifier learned on the basis of D_i
  3. *Err*$(y_i, D \setminus D_i) = \dfrac{|\{(\mathbf{x}, c(\mathbf{x})) \in D \setminus D_i : y_i(\mathbf{x}) \neq c(\mathbf{x})\}|}{|D \setminus D_i|}$

Bootstrapped misclassification rate:

$$\textit{Err}_{bt}(y, D) = \frac{1}{k} \sum_{i=1}^{k} \textit{Err}(y_i, D \setminus D_i)$$

Remarks:

- Let $|D| = n$. The probability that an example is not considered is $(1 - 1/n)^n$. Similarly, the probability that an example is considered at least once is $1 - (1 - 1/n)^n$.
- If n is large, then $1 - (1 - 1/n)^n \approx 1 - 1/e \approx 0.632$. I.e., each training set contains about 63.2% of the examples in D.
- The classifiers $y_1, \ldots, y_k$ can be used in a combined fashion, called *ensemble*, where the class is determined by means of a majority decision:

$$y(\mathbf{x}) = \underset{j \in C}{\text{argmax}}\, |\{i \in \{1, \ldots, k\} : y_i(\mathbf{x}) = j\}|$$

- For the construction of tree classifiers, bootstrapping has been reported to improve the misclassification rate about 20% – 47% compared to a standard approach. [Breiman]

Measuring Performance

Misclassification Costs

Use of a cost measure for the misclassification of a feature vector $\mathbf{x}$ in class c' instead of in class c :

$$\textit{cost}(c' \mid c) \begin{cases} \geq 0 & \text{if } c' \neq c \\ = 0 & \text{otherwise} \end{cases}$$

Estimation of $\textit{Err}^*_{\textit{cost}}(y)$ based on a sample $D_s \subseteq D$:

$$\textit{Err}_{\textit{cost}}(y, D_s) = \sum_{(\mathbf{x}, c(\mathbf{x})) \in D_s} \textit{cost}(y(\mathbf{x}) \mid c(\mathbf{x}))$$

Remarks:

- The misclassification rate *Err* is a special case of Err_{cost} with $cost(c' \mid c) = 1$ for $c' \neq c$.