

Machine/Statistical Learning Model Assesment

SICOM, M2 Sigma

2022-23

Performance criterion

We seek a regression function/classification rule f for **accurate** predictions $f(X)$ of target Y given X .

Problem : **accurate ??**

Solution : Introduction of a loss function :

$$L : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}^+,$$

in order to penalize the prediction errors :

$$L(Y, f(X)) \begin{cases} = 0 & \text{if } f(X) = Y, \\ \geq 0 & \text{otherwise,} \end{cases}$$

Performance criterion (Cont'd)

Two useful loss functions

1. Quadratic cost : $L(Y, f(X)) = (Y - f(X))^2$,
2. 0 – 1 cost :

$$L(Y, f(X)) = \begin{cases} 0 & \text{if } |Y - f(X)| < \epsilon \quad (\epsilon > 0 \text{ arbitrarily small}), \\ 1 & \text{otherwise} \end{cases}$$

Thus for a classification problem, $L(Y, f(X)) = 0$ if $Y = f(X)$, 1 otherwise

Remark

These two loss functions are the most popular. But other loss functions exist and may be useful depending on the prediction method (e.g., *cross-entropy* to train neural nets, or *hinge loss* for SVM)

Error rate

True error/Test error

The *true error rate* w.r.t. a prediction function f and a loss function L is defined as

$$\begin{aligned}\mathcal{E}[f] &= E_{X,Y} [L(Y, f(X))], \\ &= \int L(y, f(x)) dP(x, y),\end{aligned}$$

where $P(x, y)$ is the joint probability measure of (X, Y) .

👉 measure the average error rate for **new data independent from f** , aka **Test error**.

Training error

The *training error rate* w.r.t. a prediction function f , a loss function L , and a training dataset $(X_1, Y_1), \dots, (X_N, Y_N)$ is defined as the empirical mean

$$\hat{\mathcal{E}}_N[f] = \frac{1}{N} \sum_{i=1}^N L(Y_i, f(X_i))$$

👉 in practice f is learned and may over-fit the training set : **not a faithful estimator of the True error !**

Error rates for usual loss functions

Quadratic cost (rather for regression)

- True error rate → Mean Squared Error (MSE)

$$\mathcal{E}[f] = E_{X,Y} [(Y - f(X))^2] = \int (y - f(x))^2 dP(x, y),$$

- Training error rate → Residual Sum of Squares(RSS)

$$\hat{\mathcal{E}}_N[f] = \frac{1}{N} \sum_{i=1}^N (Y_i - f(X_i))^2$$

Error rates for usual loss functions (Cont'd)

0 – 1 cost (rather for classification)

For a classification problem,

- True error rate :

$$\begin{aligned}\mathcal{E}[f] &= \Pr(|Y - f(X)| > \epsilon), \\ &= \Pr(Y \neq f(X)) \leftarrow \text{misclassification probability}\end{aligned}$$

- Training error rate :

$$\begin{aligned}\hat{\mathcal{E}}_N[f] &= \frac{1}{N} \sum_{i=1}^N 1_{|Y_i - f(X_i)| > \epsilon}, \\ &= \frac{1}{N} \sum_{i=1}^N 1_{Y_i \neq f(X_i)} \leftarrow \text{misclassification rate}\end{aligned}$$

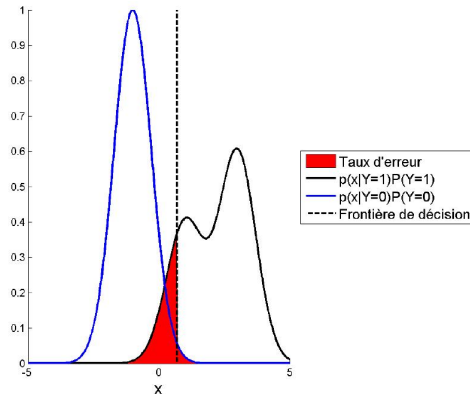
Exercise : True error rate

Consider a binary classification problem $Y \in \{0, 1\}$ with scalar data $X \in \mathbb{R}$.

For a prediction rule $f_1(X) = 0$ if $X \leq t$, 1 otherwise, where t is a given threshold, and based on the

- class weights $\Pr(Y = 0)$, $\Pr(Y = 1)$
- class pdf's $p(x|Y = 0)$, $p(x|Y = 1)$

how can we compute the true error rate (0–1 cost) ?



Optimal prediction for the regression problem (quadratic cost)

- ▶ $Y \in \mathcal{Y} \subset \mathbb{R}$ ← continuous domain

Definition

The conditional expectation of Y given $X = x$ is defined as :

$$E[Y|X = x] = \int_{\mathcal{Y}} y dP(Y|X = x) = \int_{\mathcal{Y}} y p(y|X = x) dy,$$

and we denote as $E[Y|X]$ the associated random variable.

Theorem

The regression function defined as the conditional expectation $f_{\text{MMSE}}(x) = E[Y|X = x]$ is optimal in the quadratic loss sense : for any rule f , $\mathcal{E}[f] \geq \mathcal{E}[f_{\text{MMSE}}]$.

Remarks

- ▶ $f_{\text{MMSE}}(X) \equiv$ *Minimum Mean Squared Errors* (MMSE) estimator
- ▶ In real-word applications, the distribution of (X, Y) is unknown \Rightarrow no analytical expression of $f_{\text{MMSE}}(X)$. But useful reference on academic examples.

Optimality of the Conditional Expectation

Proof

$$\begin{aligned}\mathcal{E}[f] &= E_{X,Y} [(Y - f(X))^2] = \int_{\mathcal{X} \times \mathcal{Y}} (y - f(x))^2 p(x, y) dx dy, \\ &= \int_{\mathcal{X}} \underbrace{\int_{\mathcal{Y}} (y - f(x))^2 p(y|x) dy}_{A_x} p(x) dx.\end{aligned}$$

To minimize $\mathcal{E}[f]$, it is sufficient to get a pointwise minimization of $A_x \forall x \in \mathcal{X}$:

$$\begin{aligned}A_x &= E_{Y|X=x} [(Y - f(x))^2 | X = x] = E_{Y|X=x} [(Y - E[Y|X = x] + E[Y|X = x] - f(x))^2 | X = x], \\ &= E_{Y|X=x} [(Y - E[Y|X = x])^2] + E_{Y|X=x} [(E[Y|X = x] - f(x))^2] \\ &\quad + \underbrace{2 E_{Y|X=x} [(Y - E[Y|X = x])(E[Y|X = x] - f(x))]}_{B_x}.\end{aligned}$$

But

$$\begin{aligned}B_x &= E_{Y|X=x} [(Y - E[Y|X = x])(E[Y|X = x] - f(x)), \\ &= (E[Y|X = x] - E[Y|X = x])(E[Y|X = x] - f(x)) = 0,\end{aligned}$$

thus $A_x = E_{Y|X=x} [(Y - f_{\text{MMSE}}(x))^2] + (f_{\text{MMSE}}(x) - f(x))^2 \geq E_{Y|X=x} [(Y - f_{\text{MMSE}}(x))^2]$.

Finally $\mathcal{E}[f] = E_X[A_X] \geq E_X[E_{Y|X}[(Y - f_{\text{MMSE}}(X))^2]] = \mathcal{E}[f_{\text{MMSE}}]$

□

Bayes classifier

- $Y \in \mathcal{Y}$ ← discrete domain

Definition

The Bayes classification rule f^* is defined as

$$f^*(x) = \arg \max_{k \in \mathcal{Y}} \Pr(Y = k | X = x).$$

The associated error rate $\mathcal{E}[f^*]$ is referred to as the **Bayesian error rate**

Theorem

The Bayes classification rule f^* is optimal in the 0 – 1 loss sense : for any rule f , $\mathcal{E}[f] \geq \mathcal{E}[f^*]$.

Remarks

- $f^*(X) \equiv$ *maximum a posteriori* (MAP) estimate
- In real-word applications, the distribution of (X, Y) is unknown \Rightarrow no analytical expression of $f^*(X)$. But useful reference on academic examples.

Optimality of the Bayes classifier

Proof

$$\mathcal{E}[f] = \Pr(Y \neq f(X)) = \int_{\mathcal{X}} \underbrace{\Pr(Y \neq f(X) | X = x)}_{A_x} p(x) dx,$$

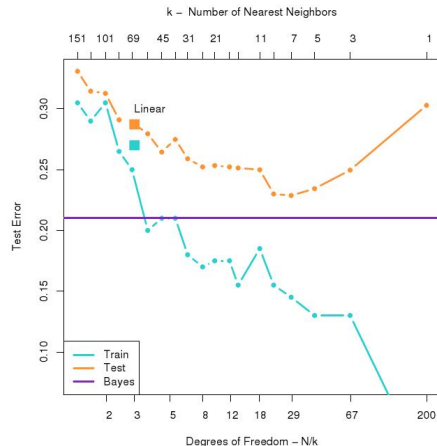
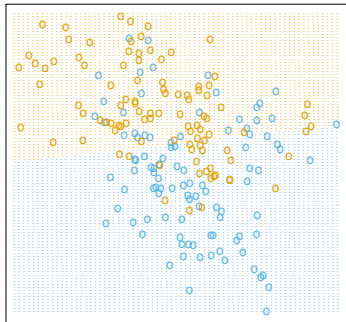
where

$$\begin{aligned} A_x &= \Pr(Y \neq f(x) | X = x), \\ &= 1 - \Pr(Y = f(x) | X = x). \end{aligned}$$

To minimize $\mathcal{E}[f]$, it is sufficient to get a pointwise minimization of A_x for $x \in \mathcal{X} \Leftrightarrow$ equivalent to maximize $\Pr(Y = f(x) | X = x)$



Toy example : Bayes classifier



Train and test errors (misclassification rate) for the binary classification example : k-NN (curves as a function of k) and linear regression (square markers)

- purple horizontal line is the Bayes classifier error, i.e. the lower bound for test error : whatever the algorithm and the size of the training set, can not do better than ≈ 0.21 !

Exercise

Statement

We consider the following binary classification problem :

- ▶ $Y \in \{0, 1\}$, $X \in [0, 5]$, $\Pr(Y = 0) = \Pr(Y = 1) = \frac{1}{2}$, and

$$X|Y = 0 \sim \mathcal{U}([0, 2]),$$

$$X|Y = 1 \sim \mathcal{U}([1, 5])$$

1. Compute the true error rates for the following classifiers :
 - ▶ $f_1(X) = 0$ if $X \leq 0$, 1 otherwise
 - ▶ $f_2(X) = 0$ if $X \leq 2$, 1 otherwise
 - ▶ $f_3(X) = 0$ if $X \leq 4$, 1 otherwise
2. Find the Bayes classifier expression and its error rate.

Exercise

Sketch of correction

1. Applying, the law of total probability, it comes that

- ▶ $\Pr(Y \neq f_1(X)) = \frac{1}{2}$
- ▶ $\Pr(Y \neq f_2(X)) = \frac{1}{8}$
- ▶ $\Pr(Y \neq f_3(X)) = \frac{3}{8}$

2. Bayes formula provides the expression of $\Pr(Y = y|X = x)$ w.r.t. the exercise statements :

$$\Pr(Y = y|X = x) = p(x|Y = y) \frac{\Pr(Y = y)}{p(x)},$$

where $p(x)$ can be computed by the law of total probability :

$$p(x) = p(x|Y = 0) \Pr(Y = 0) + p(x|Y = 1) \Pr(Y = 1).$$

By paying attention to the support of each of the conditional distributions, it comes that the Bayes classifier is $f^* \equiv f_2$, thus $\mathcal{E}[f^*] = \mathcal{E}[f_2] = \frac{1}{8}$

Recap on Train and Test Errors

- ▶ Loss-function
 - ▶ Classification : $L(y, \hat{y}) = 0$ if $y = \hat{y}$ else 1 \leftarrow 0-1 loss
 - ▶ Regression : $L(y, \hat{y}) = (y - \hat{y})^2 \leftarrow$ quadratic loss
- ▶ **Train error** : average loss over the training sample

$$\text{Err}_{\text{train}} = \frac{1}{n} \sum_{i=1}^n L(y_i, \hat{y}_i)$$

- ▶ **Test error** : average loss over a new test sample \rightarrow Generalization error
- ▶ General picture :

$$\text{Err}_{\text{test}} \approx \text{Err}_{\text{train}} + O$$

O would be the average *optimism* (overfitting problem !)

Model Assessment/Validation

Objective

Estimate the generalization error, i.e. the predictive performance on unseen/test data

Common Applications

- ▶ Hyperparameter tuning : estimate the best set of hyperparameters
- ▶ **Model selection** : compare the performance of different models

Validation procedures

Validation set

Create a validation set, i.e. fresh data not used during the training set, to estimate the predictive performances/test error rate

- ▶ Best solution : collect a large data set of new test data... But in practice, this is generally not available/possible !
- ▶ Simple solution : split the data set. One part is removed from the training set and will be kept as a validation set only



But

- ▶ only part of the data is used to train the model (introduces a bias)
- ▶ the role of the data is not symmetrical : some are used only for training, others only for testing (increases the variance)
- ▶ standard solution : [cross-validation](#), e.g. K-fold Cross-Validation

K-fold Cross-Validation (CV) : Principle

- ▶ Method to estimate prediction error using the only training sample
- ▶ Based on splitting the data in K -folds, here $K = 5$:

$\text{Err}_1(\hat{f}_1, \lambda)$	Validation	Train	Train	Train	Train
$\text{Err}_2(\hat{f}_2, \lambda)$	Train	Validation	Train	Train	Train
$\text{Err}_3(\hat{f}_3, \lambda)$	Train	Train	Validation	Train	Train
$\text{Err}_4(\hat{f}_4, \lambda)$	Train	Train	Train	Validation	Train
$\text{Err}_5(\hat{f}_5, \lambda)$	Train	Train	Train	Train	Validation

where λ are some hyperparameters of the model/method

- ▶ Estimate of Test error :

$$\text{CV}(\hat{f}, \lambda) = \sum_{k=1}^K \text{Err}_k(\hat{f}_k, \lambda)$$

K-fold Cross-Validation (CV) : Algorithm

Input : input variables X (dimension $n \times p$), responses y (dim. n), number of folds k

Divide randomly the set $\{1, 2, \dots, n\}$ in k subsets (i.e., folds) of roughly equal sizes (e.g., size equals to the integer part of n/k with a little smaller last part if n is not a multiple of k) denoted as F_1, \dots, F_k

for $i = 1$ to k :

- ▶ Form the validation set (X_{val}, y_{val}) where the indexes of the X and y variables belongs to the i th fold F_i
- ▶ Form the training set (X_{train}, y_{train}) where the indexes of the X and y variables belongs to all the folds except F_i
- ▶ Train the algorithm/model on the training set (X_{train}, y_{train})
- ▶ Apply the resulting prediction rule on the input X_{val} of the validation set
- ▶ Compute the error rate on the validation set based on the predictions and the true responses y_{val}

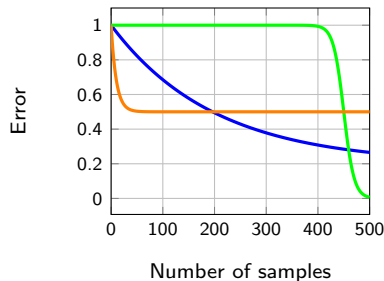
Output : the average error rate computed over all the k folds

Practical advices

- K ? Usually $K=5$ or 10 is a good trade-off ($K=n$ is called leave-one-out)

	Bias	Variance
K low	High	Low
K high	Low	High
$K = n$	Low	Very High

- Be careful to the learning curve



- Model should be trained completely for each fold (i.e., data normalization, optimization, etc ...)
- **Notebooks : `N1_validation_and_model_selection.ipynb`**

Conclusions




Model assesment methods are an essential tool for data analysis, especially for big datasets involving many predictors

Cross-validation

- ▶ generic and simple procedure that can be used for any supervised problem
- ▶ provides a direct estimate of the test error.
- ▶ Can be used for model selection and/or hyperparameter tuning
- ▶ K -fold (with $K = 5$, or $K = 10$) is a standard choice, but there exists many variants depending on the problem, e.g.
 - ▶ *stratified* K -fold to ensure that all the folds have roughly the same average response value ← useful for classification to be sure that each fold contains roughly the same proportions of class labels.
 - ▶ *hold-out* cross-validation for time-series where a subset (split temporally) of the data is reserved for validating the model performance

Supplementary materials

Cross-validation

-  Coursera MOOC short (12mn) video <https://fr.coursera.org/lecture/machine-learning/model-selection-and-train-validation-test-sets-QGKbr>
-  Wikipedia page [https://en.wikipedia.org/wiki/Cross-validation_\(statistics\)](https://en.wikipedia.org/wiki/Cross-validation_(statistics))
-  Scikit-learn documentation (with examples) :
 - ▶ https://scikit-learn.org/stable/modules/cross_validation.html#
 - ▶ https://scikit-learn.org/stable/modules/grid_search.html#grid-search