Resampling methods for error rate estimation in supervised learning

Ricco RAKOTOMALALA
1. Error rate estimation
2. Resubstitution error rate
3. Holdout approach
4. Cross-validation
5. Bootstrap
6. Influence of the sampling scheme
Measuring the performance of the classifiers

The inability to measure the true error rate on the whole population

**Starting point:** We have a sample of size "n" as from which we want to build a classifier $M(n)$

\[ \hat{Y} = M(X, n) \]

**Prediction error rate:** The "true" error rate can be obtained by the comparison of the observed values of $Y$ and the prediction of the classifier $M$ on the whole population.

\[ \mathcal{E} = \frac{\sum_{\omega \in \Omega_{pop}} [Y(\omega) \neq \hat{Y}(\omega)]}{\text{card}(\Omega_{pop})} \]

But:

1. The "whole" population is never available
2. Accessing to all the instances is too costly

How to do by having in everything and for everything the sample of size "n" to learn the model and to measure its performance...
Measuring the performance of the classifiers
Illustration with the "waves" dataset – Breiman and al. (1984)

Description:
• One target variable (3 classes of waves) and 21 continuous predictive attributes
• Generated dataset - Potentially of infinite size
• n = 500 instances, used for the learning process
• n = 500,000 instances, the “population” used for measuring the “true” error rate (baseline measure)
• 3 learning algorithms (LDA, C4.5 and Perceptron) which have various behaviors

The “true” error rate: measured on the “population” (500,000 instances)

<table>
<thead>
<tr>
<th></th>
<th>Erreur &quot;théorique&quot; (Calculé sur 500000 obs.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDA</td>
<td>0.185</td>
</tr>
<tr>
<td>C4.5</td>
<td>0.280</td>
</tr>
<tr>
<td>RNA (10 CC)</td>
<td>0.172</td>
</tr>
</tbody>
</table>

In practice, we have never an unlimited number of instances. Thus, we must use the available sample (n = 500) instances in order to learn the model and estimate its error rate. For each classifier, the estimated error rate must be as close as possible to the "true" value above.
Resubstitution error rate

Use the same dataset for the learning phase and the evaluation phase

Steps:
• Learn the classifier on the sample (n= 500)
• Apply the classifier on the same sample
• Build the confusion matrix and calculate the error rate
  → This is the resubstitution error rate.

$$e_r = \frac{\sum_{\omega \in \Omega} [Y(\omega) \neq \hat{Y}(\omega)]}{n}$$

Results

<table>
<thead>
<tr>
<th></th>
<th>Erreur &quot;théorique&quot;</th>
<th>Erreur Resubstitution</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDA</td>
<td>0.185</td>
<td>0.124</td>
</tr>
<tr>
<td>C4.5</td>
<td>0.280</td>
<td>0.084</td>
</tr>
<tr>
<td>RNA (10 CC)</td>
<td>0.172</td>
<td>0.064</td>
</tr>
</tbody>
</table>

Comments:
• The resubstitution error rate underestimates very often the true error rate
• The gap depends on the characteristics of the dataset AND classifier
• More a point influences its own affectation, more the optimism bias will be high
  → (1) NN, 1-NN: resubstitution error rate = 0% is possible, etc.
  → (2) Classifiers with high complexity
  → (3) Small sample size (n is low)
  → (4) High dimensionality (in relation to the sample size) and noisy variables
Behavior of the resubstitution error rate (blue) and the true error rate (purple) according to the complexity of the classifier and the sample size.

The algorithm begins to learn sample-specific "patterns" that are not true to the population (e.g. too many variables, too many neurons in the hidden layer, too large decision tree...).

The larger is the sample size, the more efficiently we learn the "underlying relationship" between X and Y in the population.

The larger is the sample size, the less is the dependence of the algorithm to the sample singularities.
The holdout approach
Split the dataset into train sample and test sample

Learning phase, train set

$$\Omega_a$$  
$$\approx 60\% \sim 70\%$$

Testing phase, test set

$$\Omega_t$$  
$$\approx 30\% \sim 40\%$$

$$M(X, n_a)$$

$e_t = \frac{\sum_{\omega \in \Omega_t} [Y(\omega) \neq \hat{Y}(\omega)]}{n_t}$

**Test error rate**
Unbiased estimation of the $M(X, n_a)$ error rate

Modèle : LDA($X, 300$)  \( \Rightarrow \)  \( \varepsilon = 0.2099 \) Computed on the 500,000 instances

Test set : 200 obs.  \( \Rightarrow \)  \( \varepsilon = 0.1950 \)

**Experiments**
Repeat 100 times the process
300 inst. train, 200 inst. test  \( \Rightarrow \)
The holdout approach
Bias and variance

\[ e_t \] is an **unbiased** estimation of the error rate of \[ M(X, n_a) \]

**But** This is a **biased** estimation of the error of \[ M(X, n) \]

LDA(X, 300)  
LDA(X, 500)

Part of the data only (300 inst.) is used to learn the model, the learning is of lower quality than if we use the whole sample with \( n = 500 \) inst.

The "bias" is lower when the train sample is larger.

The test error rate is accurate when the test sample size is high. The larger is the test sample, the lower is the variance.

Large train set and large test set are not compatible.
The holdout approach

Experiments

“True” error rate of LDA(X,500) = 0.185

Conclusion:
• The test error rate is an unbiased estimation of the performance of the classifier learned on the train sample.
• But it is a bad estimation of the performance of the classifier learned on the whole dataset
• The holdout approach is only interesting when we handle large database
• Otherwise, we are facing a dilemma: increase the train sample size to obtain a good model but bad error evaluation, or increase the test sample size to obtain a better error rate estimation of a bad model.
How to estimate the error of $M(X,n)$?

Resampling techniques

Cross-validation
Leave-one-out
Bootstrap
Cross-validation Principle

**Algorithm**

- Subdivide the sample in $K$ folds (groups) – $n_k$ is the size of the $k^{th}$ fold
- For each $k$:
  - Construct the model $M(X, n-n_k)$
  - Calculate its test error rate on $n_k \rightarrow e_k$
- $e_{cv}$ = the mean of the errors $e_k$

“True” error rate of LDA($X,500$) = 0.185

- $K=10$ gives a good compromise between “bias” and “variance” for the most of the situations (dataset and learning algorithm)

- Repeated cross-validation may improve its characteristics ($B \times K$-Fold Cross validation)

- In the case of overfitting, the cross-validation (especially when $K$ is high) tends to underestimate the true error rate
**Leave-one-out**

Special case of cross-validation where $K = n$

---

**Algorithm**

- Subdivide the sample into $K=n$ folds
- For each instance $k$:
  - Construct the classifier $M(X,n-1)$
  - Apply the classifier on the $k^{th}$ instance $\rightarrow e_k$
- Calculate the mean $e_{loo}$ of the errors

- Significantly more computationally expensive than the $K (K << n)$ cross validation without being best
- Dramatically underestimate the error rate in the case of overfitting

---

### Error Results

<table>
<thead>
<tr>
<th>Method</th>
<th>10-CV</th>
<th>Leave one out</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDA</td>
<td>0.170</td>
<td>0.174</td>
</tr>
<tr>
<td>C4.5</td>
<td>0.298</td>
<td>0.264</td>
</tr>
<tr>
<td>RNA (10 CC)</td>
<td>0.174</td>
<td>0.198</td>
</tr>
</tbody>
</table>

---

Experiments (with the sample of size 500)

We can decrease the variance by repeating the process

Only one measurement is possible on a sample of size $n$. 

$e_k = 1$ (error) or $0$ (good prediction)
Bootstrap Principle

Algorithm

- Repeat B times (called “replications”)
  - Sample with replacement a dataset of size n → \( \Omega_b \)
  - Separate the unselected instances → \( \Omega_{(b)} \)
  - Construct the classifier with the dataset \( \Omega_b \)
  - Calculate the resubstitution error rate on \( \Omega_b \) \([e_r(b)]\)
  - Calculate the test error rate on \( \Omega_{(b)} \) \([e_t(b)]\)
  - Calculate the “optimism” \( o_b \)

On the whole dataset (size n), calculate the resubstitution error rate

\[
e_B = e_r + \frac{\sum_{b} o_b}{B}
\]

1. \( e_r \) is the resubstitution error rate
   - The bootstrap enables to estimate the "optimism"
   - It is used to correct the resubstitution error rate
   - The correction is often a little excessive
     (the error is often overestimated with the standard bootstrap)

2. \( e_{0.632B} = 0.368 \times e_r + 0.632 \times \frac{\sum_{b} e_t(b)}{B} \)
   - 0.632 bootstrap
     → Weight with the probability of belonging to \( \Omega_b \) for a replication (#0.632)
     - The correction is more realistic

3. It exists another approach which allows to correct the "optimism" by taking account the classifier characteristic: 0.632+ bootstrap
Bootstrap
Experiments -- 0.632 Bootstrap

“True” error rate
LDA = 0.185

Increase the number of replication

Decreasing of the variance – B # 25 is enough
Little influence on the bias

The bias comes from the fact that, with each replication, n instances well used to construct the model, but as some instances are repeated, the information is redundant, the model is less efficient... we cannot modify this behavior...
Cross-validation or bootstrap?

At equal cost calculation (here 10 repetitions of the train-test operations)

- Bootstrap has lower variance
- But the cross-validation is less biased (we have a counter-example here, but it is only a simulation with 100 repetitions on one dataset and one kind of classifier)

In scientific publications, researchers seem to favor cross-validation ... maybe because it is easier to implement.
Sampling scheme

Influence of the sampling method on the organization of the cross-validation
Stratified sampling, cluster sampling, etc.

**General principle:** the mode of constitution of the folds must respect the sampling method for the constitution of the dataset

→ If the dataset comes from a stratified sampling from the population, we must use the same way for the selection of instances in each fold e.g. defining the same proportion of the classes in each fold

Goal: decrease the variance

→ If the dataset comes from a cluster sampling, the sample unit becomes the clusters in order to constitute the folds

Goal: decrease the bias
Conclusion

• Resubstitution error is (almost) always too optimistic i.e. underestimates the true error rate
• This optimism depends on the characteristics of the data and the classifier
• The holdout approach is only interesting on large dataset (number of instances)
• The test error estimates the performance of the classifier learned on the train sample
• But it gives a bad indication about the performance of the classifier learned from the whole dataset
• Cross-validation and bootstrap are equivalent in general
• K = 10 seems a good compromise for the cross-validation
• Repeated cross-validation decreases the variance, but not in a spectacular way
• 0.632 bootstrap has a lower variance than the cross-validation, but higher bias
• We cannot really correct this bias (the 0.632+ can handle this but it is not very common in software)
• In the case of overfitting, both cross-validation and bootstrap cannot give a good estimation of the error rate
References
